Technische Universität Braunschweig

INSTITUT FÜR KONSTRUKTIONSTECHNIK

Carsten Stechert

Modellierung komplexer Anforderungen

Bericht Nr. 75, 2010

Modellierung komplexer Anforderungen

Von der Fakultät für Maschinenbau
der Technischen Universität Carolo-Wilhelmina
zu Braunschweig

zur Erlangung der Würde eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte

Dissertation

von

Dipl.-Ing. Carsten Stechert
aus Celle

Eingereicht am:	25.02.2010
Mündliche Prüfung am:	01.06.2010
Referenten:	Prof. Dr.-Ing. H.-J. Franke
	Prof. Dr.-Ing. H. Birkhofer
Vorsitzender:	Prof. Dr.-Ing. T. Vietor

2010

Vorwort

Diese Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Institut für Konstruktionstechnik der Technischen Universität Carolo-Wilhelmina zu Braunschweig.

Herr Prof. Dr.-Ing. Hans-Joachim Franke hat in seiner Zeit als Institutsleiter eine Arbeitsumgebung geschaffen, die es mir ermöglichte eigenverantwortlich Projekte zu bearbeiten und neue Ideen umzusetzen. Für seinen Rat und für die Betreuung meiner Dissertation danke ich ihm sehr. Herrn Prof. Dr. h.c. Dr.-Ing. Herbert Birkhofer danke ich für die Übernahme des Koreferats und sein Interesse an meiner Arbeit. Herrn Prof. Dr.-Ing. Thomas Vietor danke ich für die Übernahme des Prüfungsvorsitzes und dafür, dass er mir als Institutsleiter in der Endphase meiner Arbeit Freiraum ließ die Arbeit zu einem guten Abschluss zu bringen.

Außerdem danke ich allen Mitarbeitern des IK für die gute und kollegiale Zusammenarbeit. Insbesondere danke ich Frau Ursula Gent für die Gestaltung von Abbildungen und Herrn Dipl.-Ing. David Inkermann für die Durchsicht des Manuskripts. Vielen Dank auch an die Kollegen aus dem SFB 562 und an alle studentischen Hilfskräfte, Studien- und Diplomarbeiter und „Versuchskaninchen“ aus der Vorlesung RAO, die mich in meiner Zeit am Institut unterstützt haben.

Ein besonderer Dank gilt meinen Eltern, für ihre ständige Unterstützung. Zum Schluss möchte ich Christin für ihre Hilfe und ihre Liebe danken und dafür, dass sie mich immer mal wieder daran erinnerte, dass es auch Zeiten im Leben ohne Arbeit geben muss.

Braunschweig im Juli 2010 Carsten Stechert

Inhaltsangabe

Die moderne Produktentwicklung wird immer komplexer und komplizierter, da Produkte verstärkt interdisziplinären Charakter aufweisen, eine hohe Variantenvielfalt gefordert wird und die geforderten Markteinführungszeiten immer kürzer werden. Um diesen Herausforderungen zu begegnen, werden Produkte heutzutage häufig in Kooperationsnetzwerken entwickelt und nach Baukastenprinzipen gestaltet. Durch den sich immer weiter verschärfenden Konkurrenzdruck kommt der systematischen Erfüllung der Kundenwünsche auf der einen und der zielgerichteten Erreichung der Unternehmensziele auf der anderen Seite eine entscheidende Bedeutung zu.

Ziel der vorliegenden Arbeit ist es, ein Modellierungskonzept zu entwickeln, dessen Kern die Erfassung, Verarbeitung und Bereitstellung von Anforderungen bildet. Die Modellierung soll sich den modernen Herausforderungen der Produktentwicklung stellen und einen ganzheitlichen Ansatz ermöglichen.

Zunächst werden die Herausforderungen an eine moderne Produktentwicklung systematisch identifiziert. Als Schwerpunkte werden Produktentwicklungsprozesse verschiedener Fachbereiche, die Schwierigkeiten verteilter Produktentwicklung und die besonderen Herausforderungen bei der Entwicklung von Baukastensystemen betrachtet. Dann werden verschiedene Möglichkeiten der Modellierung untersucht und grundlegende Aussagen zu relevanten Partialmodellen, Beschreibungssprachen und Auswertemechanismen getroffen. Schließlich wird auf Basis der Systems Modelling Language (SysML) ein umfassendes Modellierungskonzept erarbeitet, mit dessen Hilfe das Produkt auf Systemebene ganzheitlich betrachtet werden kann. Durch die entwickelten Rechnerhilfsmittel wird eine zielgerichtete Auswertung der Modelle ermöglicht. Dadurch kann das Anforderungsmodell deutlich verbessert und systematisch Zielkonflikte identifiziert werden. Schließlich wird ein Vorgehen vorgestellt, um mit Hilfe des erarbeiteten Konzepts zielgerichtet Baukastensysteme in der interdisziplinären, verteilten Produktentwicklung zu erarbeiten. Anhand einer Beispielaufgabe wird im Rahmen einer Vorlesung die Akzeptanz des Konzepts bei der praktischen Anwendung untersucht. Abschließend wird die Anwendbarkeit des Modellierungskonzepts auf komplexere Problemstellungen im Rahmen des Sonderforschungsbereich 562 „Robotersysteme für Handhabung und Montage“ überprüft.

Abstract

Modern product development is getting more and more complex and complicated. This is due to a more and more interdisciplinary character of products, a high demanded variant diversity and short expected market introduction time. To cope with these challenges products are often developed in cooperation networks and designed following modular approaches. The tightening competition pressure induces the need for systematic fulfilment of customer wishes and purposive achievement of enterprise goals.

This work aims towards a modelling concept, whose core is made of gathering, processing, and providing requirements. The resulting models shall face the modern challenges of product development and allow an holistic approach.

Initially, challenges on modern product development are systematically identified. The analysis focuses on product development processes of different disciplines, difficulties of distributed development, and particularities of modular system development. Then different possibilities of modelling are analyzed and basic information of relevant partialmodells, modelling languages and evaluation mechanisms is worked out. Afterwards, a comprehensive modelling concept basing on the Systems Modelling Language (SysML) is developed that allows considering the product holistically. With the help of the developed computer aided tools a goal-oriented model evaluation is possible. The requirements model can be considerably improved and goal conflicts can be systematically identified. A process is proposed that uses the developed concept for a goal-oriented elaboration of modular systems within an interdisciplinary, distributed design process. The acceptance of the developed concept in practical use is examined by means of a sample problem within a lecture. Eventually, the practicality for a complex problem and product is proved in the scope of the collaborative research project „Robotic Systems for handling and Assembly" (SFB 562).

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Motivation 1
1.2 Aufbau der Arbeit 3

2 Stand der Forschung in der interdisziplinären Produktentwicklung **5**
2.1 Produktentwicklungsprozesse 6
2.1.1 Systemidee beschreiben 8
2.1.2 Situation analysieren 9
2.1.3 Ziele formulieren 10
2.1.4 Projekt planen und überwachen 11
2.1.5 Systemkontext beschreiben 12
2.1.6 Fachwissen und Entscheidungen dokumentieren 13
2.1.7 Anforderungen erfassen 14
2.1.8 Anforderungen bereitstellen 15
2.1.9 Anforderungen verarbeiten 17
2.2 Verteilte Produktentwicklung 18
2.2.1 Kommunikation zwischen Entwicklungspartnern 19
2.2.2 Austausch von Informationen 20
2.2.3 Arbeitsteilung 21
2.2.4 Kombination und Integration von Ergebnissen 22
2.2.5 Menschliches Verhalten und ihre Beziehungen 22
2.2.6 Methoden und Werkzeuge 24
2.2.7 Organisatorische Aspekte 26
2.3 Modularität und Flexibilität von Produkten 26
2.3.1 Anforderungen in der Baukastenentwicklung 29
2.3.2 Änderungen in Produktentwicklung und Betrieb 32
2.4 Bewerten und Testen 34

3 Modellierung im Produktentwicklungsprozess **37**
3.1 Anforderungen an Modelle 39
3.2 Verschiedene Partialmodelle der Produktentwicklung 41
3.3 Beschreibungssprache und Werkzeug 45

3.3.1 Die Systems Modeling Language (SysML) 45
3.3.2 SysML Entwicklungswerkzeuge 47
3.4 Auswertung 50
3.4.1 Klassifikation 50
3.4.2 Beziehungen 52
3.4.3 Kennzahlen 63
3.4.4 Diagramme 66
3.4.5 Tabellen und Matrizen 67

4 Konzept zur Anforderungsmodellierung 69
4.1 Verwendete Partialmodelle 72
4.1.1 Systemideemodell 73
4.1.2 Zielmodell 75
4.1.3 Produktlebenslaufmodell 76
4.1.4 Produktumgebungsmodell 78
4.1.5 Stakeholdermodell 79
4.1.6 Anforderungsmodell 81
4.1.7 Systemkontextmodell 86
4.1.8 Testmodell 89
4.2 Arbeiten mit dem Modell 96
4.2.1 Bildung von spezifischen Sichten 96
4.2.2 Stakeholder-Anwendungsfälle-Matrix 97
4.2.3 Anforderungen-Anwendungsfälle-Matrix 100
4.2.4 Anforderungen-Komponenten-Matrix 102
4.2.5 Testkriterien-Testfälle-Matrix 103
4.2.6 Anforderungen-Anforderungen-Matrix 105
4.2.7 Zielverfolgung 108
4.2.8 Projektüberwachung 109
4.3 Vorgehen zur Baukastenentwicklung 112

5 Erprobung des entwickelten Konzepts 117
5.1 Akzeptanzuntersuchung 117
5.1.1 Aufgabe 118
5.1.2 Versuchsgruppe 118
5.1.3 Versuchsdurchführung 119
5.1.4 Ergebnisse 120
5.1.5 Fazit 124
5.2 Modellierung von Parallelrobotern 125
5.2.1 Idee für ein Handhabungs- und Montagesystem 129
5.2.2 Ziele für die Entwicklung eines Parallelroboters 130
5.2.3 Produktlebenslauf eines Parallelroboters 132
5.2.4 Produktumgebung eines Parallelroboters 133
5.2.5 Stakeholder der Parallelroboterentwicklung 134

5.2.6 Anforderungen an Parallelroboter 135
5.2.7 Entwicklung des Parallelrobotersystems 140
5.2.8 Testen des Parallelroboters 144
5.3 Rekonfigurationsparameter identifizieren 148

6 Zusammenfassung und Ausblick 161

Literaturverzeichnis 165

Abbildungsverzeichnis 185

Tabellenverzeichnis 189

Glossar 191

Formelzeichen 201

A Anhang 203
A.1 Zusammenfassung der Ansätze zur Produktentwicklung 203
A.2 Eigenschaften von Anforderungen 206

B Anhang 209
B.1 Modellierung 209
B.2 Mögliche gegenseitige Beeinflussung der Partialmodelle 214

C Anhang 219
C.1 Anforderungsmanagement in der Vorlesung RAO 219

Kapitel 1

Einleitung

„Aller Dinge Maß ist der Mensch, der Seienden, dass sie sind, der nicht Seienden, dass sie nicht sind."

- Protagoras, 485-415 v. Chr. -

Vor über 2400 Jahren prägte der griechische Philosoph *Protagoras* den oben zitierten Homo-Mensura-Satz. Zwar sind Protagoras' Originalschriften nicht erhalten, verschiedene Autoren schreiben ihm jedoch die Lehre zu, es gäbe keine objektiv richtigen, sondern lediglich subjektive Wahrheiten. Das Sein des Menschen sei subjektiv und wandelbar.

Die moderne Produktentwicklung hat erkannt, dass ein Produkt nur dann erfolgreich am Markt sein kann, falls es die Bedürfnisse der Kunden erfüllt. Kaufentscheidungen werden von Menschen auf Basis von Kriterien getroffen, die bei Konsumgütern häufig subjektiven (z.B. Markenimage), bei Investitionsgütern möglichst objektiven (z.B. Produktgesamtkosten über drei Jahre) Charakter besitzen. Eine rein objektive Beurteilung der Kriterien ist in vielen Fällen allerdings nicht möglich (z.B. Bedienbarkeit einer Maschine). Zudem kann sich auf Grund geänderter Randbedingungen eine neue Einschätzung ergeben. Beispielsweise konnte die zunächst viel versprechende Concorde wegen der Ölpreiskrisen der 1970er Jahre nie wirtschaftlich betrieben werden. Aus politischen Gründen wurde sie dennoch fast dreißig Jahre im Liniendienst eingesetzt.

Anforderungen an ein Produkt hängen von dem sie stellenden Menschen und dessen spezieller Umgebung ab. Und die Anforderungen ändern sich, so wie sich die Sichtweisen der Menschen ändern, die das Produkt nutzen bzw. an der Produktentwicklung beteiligt sind. Ein gutes Produkt kann nur dann entwickelt werden, wenn der Mensch und dessen Anforderungen ins Zentrum der Entwicklung gestellt werden: Der Mensch ist das Maß der Dinge.

1.1 Motivation

Viele Produkte werden heutzutage fachbereichsübergreifend entwickelt. Vom Bereich der Konsumgüter (z.B. DVD-Spieler) bis zu den Investitionsgütern (z.B. Industrieroboter) steigt der Mechatronikanteil stetig an. Die Teillösungen der

einzelnen Fachbereiche können nicht mehr getrennt voneinander entwickelt werden, da sie einen großen gegenseitigen Einfluss ausüben. So kann z.B. durch eine geschickte Gestaltung der kinematischen Struktur eines Parallelroboters der regelungstechnische Aufwand stark reduziert werden. Es entsteht ein zunehmender Zwang zu einer verbesserten Kommunikation und frühen Zusammenarbeit zwischen den Fachbereichen.

Mechatronische Produkte werden häufig in Kooperationsnetzwerken produziert. Dabei bringen die einzelnen beteiligten Unternehmen ihr spezielles Fachwissen in das gemeinsame Produkt ein. Das Arbeiten in Kooperationsnetzwerken erzeugt jedoch auch Schwierigkeiten. Beispielsweise verschlechtert das verteilte Arbeiten an verschiedenen Standorten die Kommunikation. Außerdem muss jedes der Unternehmen für sich wirtschaftlich arbeiten. Dadurch werden nicht immer Entscheidungen getroffen, die die Herstellung eines optimalen Produkts verfolgen.

Die beteiligten Unternehmen haben ein großes Interesse daran, dass die von ihnen entwickelten Module auch in anderen Projekten und Produkten Verwendung finden. Beispielsweise versucht ein Hersteller von Kombianzeigegeräten sein Produkt in jeweils anderem Design in Fahrzeugen unterschiedlicher Automobilkonzerne einzusetzen. Gleichzeitig versuchen die Konzerne, sich nicht von einem Zulieferer abhängig zu machen. Zur Steigerung der Flexibilität und Variantenvielfalt ist ein maßgeschneidertes Baukastensystem bzgl. der jeweiligen Kundenbedürfnisse notwendig.

Letztendlich hängt der Erfolg jedes Produktes davon ab, ob es die Kundenwünsche erfüllt. Sollte das Verhalten des Produktes bzw. die Produkteigenschaften wider Erwarten anders ausfallen bzw. vom Kunden anders aufgefasst werden als erwartet, muss schnell reagiert werden. Das Produkt muss vor Verkaufsstart angepasst werden bzw. ein verbessertes Nachfolgemodell muss schnell auf den Markt gebracht werden. Das ist nur möglich, wenn die Zusammenhänge zwischen den nicht erfüllten Kundenerwartungen und den beeinflussbaren Produktmerkmalen bekannt sind.

In Abb. 1.1 sind diese Aussagen graphisch zusammengefasst. Die wohl wichtigsten Eingangsdaten in die Produktentwicklung sind die Kundenwünsche, die häufig subjektiv und wandelbar sind. Gerade deshalb müssen sie sorgfältig erfasst, bereinigt und in technische Anforderungen transformiert werden. Des Weiteren sind Randbedingungen zu berücksichtigen, die aus Normen und gesetzlichen Vorschriften resultieren können oder technische und technologische Restriktionen abbilden. Nicht zu vergessen sind außerdem die strategischen Ziele des Unternehmens bzw. bei Kooperationsnetzwerken der verschiedenen Unternehmen. Die Gesamtheit der erfassten Eingangsdaten muss für die weitere Produktentwicklung verarbeitet und gezielt bereitgestellt werden. So entstehen Anforderungssätze, die sich an das Gesamtsystem (in Abb. 1.1 der Parallelroboter) oder einzelne Komponenten (z.B. Gelenke) richten.

Trotz einiger Ansätze für eine mechatronische Produktentwicklung (z.B. VDI-

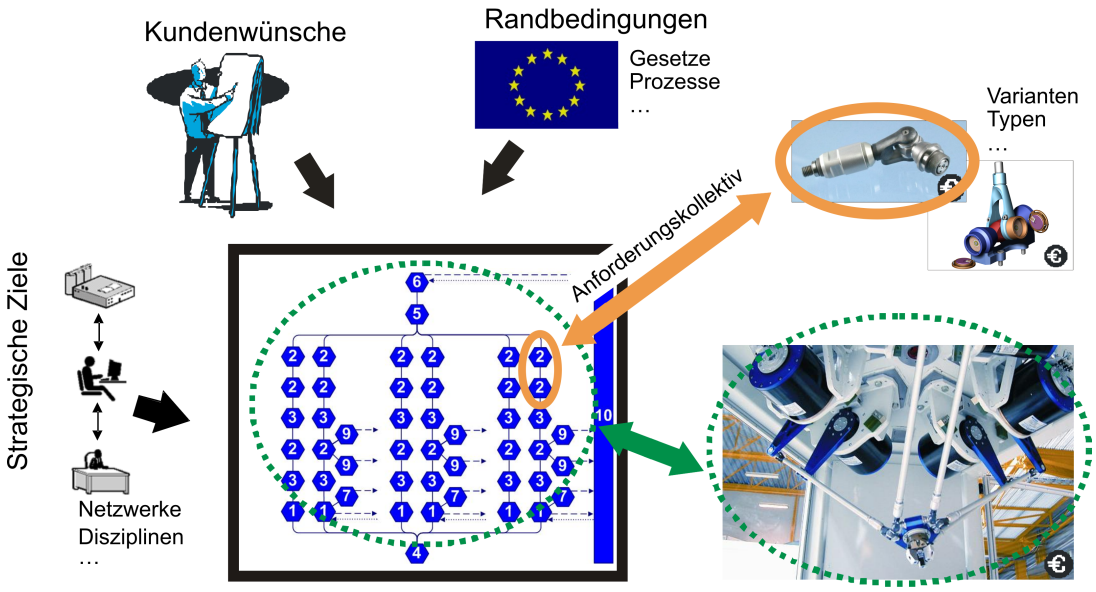


Abbildung 1.1: *Überblick über die Herkunft von Anforderungen und ihre strukturierte Verwendung für die Produktentwicklung am Beispiel eines Parallelroboters.*

Richtlinie (VDI-RL)-2206 [VDI04]) existiert bisher kein durchgängiges Vorgehen für eine interdisziplinäre Entwicklung, dass die komplexen Anforderungen ausreichend berücksichtigt. Insbesondere gibt es keine ausreichend ausgearbeitete Unterstützung der Modellierung, die als Kommunikations- und Dokumentationsgrundlage bei der Entwicklung dient.

Ziel dieser Arbeit ist es, ein einfaches Verfahren zu entwickeln, mit dessen Hilfe komplexe Anforderungen in den frühen Phasen der Entwicklung mechatronischer Produkte systematisch erfasst, aufbereitet und gezielt bereitgestellt werden können. Insbesondere soll das verteilte Arbeiten in Kooperationsnetzwerken und die Aspekte der Baukastenentwicklung berücksichtigt werden. Hierzu werden Modelle definiert und die Modellierung durch die Beschreibungssprache Systems Modelling Language (SysML) unterstützt. Das Arbeiten mit diesen Modellen soll bereits in frühen Entwicklungsphasen hinreichend gute Ergebnisse liefern und gleichzeitig relativ wenig Einarbeitung erfordern.

1.2 Aufbau der Arbeit

Der Aufbau dieser Arbeit ist in Abb. 1.2 grafisch dargestellt. Nach dieser Einleitung folgt die Wiedergabe des Stands der Technik in der interdisziplinären Produktentwicklung. In diesem Kapitel werden zunächst die verschiedenen Produktentwicklungsprozesse unterschiedlicher Fachdisziplinen untersucht und die Schwierigkeiten bei einer verteilten Produktentwicklung identifiziert. Anschlie-

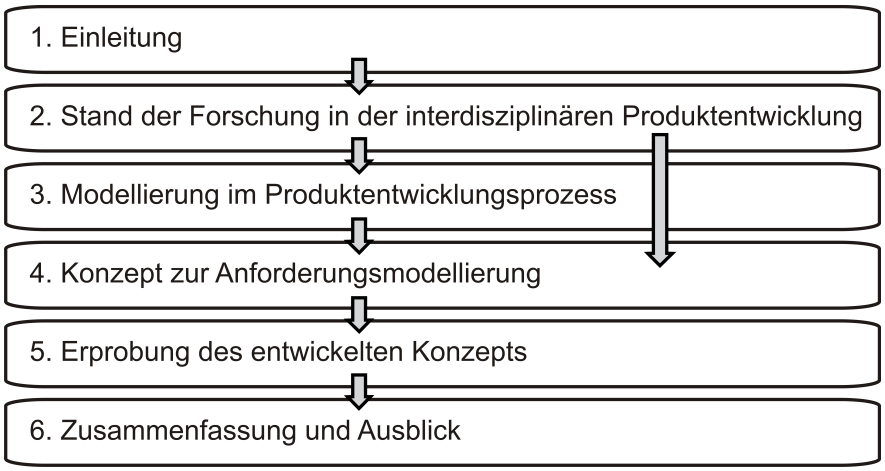


Abbildung 1.2: *Grafischer Überblick über den Aufbau dieser Arbeit.*

ßend werden die notwendigen Grundlagen zur Modularität und Flexibilität sowie zum Bewerten und Testen von Produkten zusammengestellt.

Aufbauend auf diesen Grundlagen wird im folgenden Kapitel die Modellierung im Produktentwicklungsprozess untersucht. Es werden zunächst Anforderungen an Modelle aufgestellt und dann die auf der Systemebene notwendigen Partialmodelle vorgestellt. Anschließend werden die Gründe für die Wahl der Beschreibungssprache SysML dargelegt und ihre Grundzüge sowie die der genutzten Software erklärt. Am Ende des Kapitels werden grundlegende Möglichkeiten zur Modellauswertung diskutiert, wobei besonders auf die systematische Erfassung und Verarbeitung von Beziehungen eingegangen wird.

In Kapitel 4 werden die in den beiden vorherigen Kapiteln geschaffenen Grundlagen genutzt, um ein Konzept zur Anforderungsmodellierung zu erarbeiten. Dazu werden die im Konzept verwendeten Partialmodelle genau erklärt. Anschließend folgt eine Erläuterung der rechnerunterstützten Auswertung des Modells. Abschließend wird ein Vorgehen zur Baukastenentwicklung vorgeschlagen, welches das vorgestellte Modellierungskonzept nutzt.

Im darauf folgenden Kapitel wird das Konzept anhand von Anwendungsbeispielen erprobt. In einem ersten Schritt wird die Akzeptanz des Modellierungskonzepts durch eine kleinere Entwicklungsaufgabe in einem studentischen Projekt untersucht. Im zweiten Schritt wird die Modellierung auf das komplexe mechatronische Produkt Parallelroboter erfolgreich angewendet.

Die Arbeit schließt mit der Zusammenfassung und einem Ausblick auf weitere Forschungsarbeiten. Häufig verwendete Begriffe und Abkürzungen werden im Glossar genauer beschrieben.

Kapitel 2

Stand der Forschung in der interdisziplinären Produktentwicklung

Die Entwicklung komplexer Produkte findet immer häufiger in Form einer interdisziplinären Zusammenarbeit in Kooperationsnetzwerken statt. Hierfür wird das Gesamtsystem in kleinere, handhabbare Teilsysteme (oder Teilaufgaben) zerlegt, die im Idealfall unabhängig voneinander bearbeitet werden können. Eine große Herausforderung ist es, diese Teilsysteme im Verlauf einer iterativen und dynamischen Entwicklung widerspruchsfrei zu halten. Die Gründe der Dynamik sind im Allgemeinen unscharfe Systemgrenzen und Auswirkungen von Änderungen auf mehrere Teilsysteme gleichzeitig. Darüber hinaus sollen die Produkte kostengünstig und flexibel für viele unterschiedliche Anwendungen einsetzbar sein.

Für eine systematische Untersuchung solcher Entwicklungsprozesse müssen eine Reihe ganz unterschiedlicher Bereiche betrachtet werden. In den folgenden Abschnitten werden zuerst die verschiedenen Vorgehensweisen zur Produktentwicklung in den betrachteten Fachdisziplinen untersucht und Gemeinsamkeiten für eine interdisziplinäre Entwicklung identifiziert. Dabei wird der interdisziplinäre Entwicklungsprozess in ganzer Breite betrachtet und ein Schwerpunkt auf die Anforderungsverarbeitung gesetzt.

Anschließend werden die Schwierigkeiten, die sich bei der verteilten, interdisziplinären Produktentwicklung ergeben systematisch dargestellt. Dabei werden rein technische (z.B. Datenformate, Kommunikation zwischen Standorten) aber auch menschbezogene Faktoren (z.B. Kommunikationsprobleme, Organisation) betrachtet.

Die Globalisierung und Übersättigung der Märkte führt bekanntermaßen zu einem immer stärkeren Wettbewerbs- und Kostendruck. Um ein Konzept zu erarbeiten, das die Entwicklung flexibler, modularer Produkte ermöglicht, werden in Abschnitt 2.3 die Grundzüge von Baukastensystematiken und Baukastenentwicklungsmethoden dargestellt. Dabei werden Baukastensysteme neben der Verwirklichung von Flexibilität als Mittel zur Kostenreduktion angesehen. Um die Zweckmäßigkeit der entwickelten Baukästen zu überprüfen, muss daher früh eine Kostenbetrachtung durchgeführt werden. Insbesondere bei Investitionsgütern müssen die gesamten Lebenslaufkosten berücksichtigt werden.

Zum Abschluss dieses Kapitels wird darauf eingegangen, wie Produkte und Konzepte bewertet und die Erreichung der festgelegten Ziele und Anforderungen an das Produkt überprüft werden können.

2.1 Produktentwicklungsprozesse

In den unterschiedlichen hier betrachteten ingenieurtechnischen Fachdisziplinen Maschinenbau, Elektrotechnik und Softwareentwicklung haben sich verschiedene Vorgehensmodelle etabliert, die eine zielgerichtete und effiziente Produktentwicklung fördern sollen. So sollen Zustände vermieden werden, wie sie seit den 1970er Jahren in Bildern wie Abb. 2.1 humoristisch aber treffend beschriebenen werden. Dabei wird anhand des einfachen Beispiels einer Reifenschaukel der Unterschied dessen dargestellt, was der Kunde wirklich brauchte und was von den verschiedenen Beteiligten beschrieben bzw. umgesetzt wurde. Die Einfachheit des Beispiels macht deutlich, wie wichtig eine gute methodische Behandlung der Anforderungen ist, um ein Produkt zu entwickeln, dass die wirklichen Kundenbedürfnisse erfüllt.

Die Vision der Konstruktionsmethodik der 60er und frühen 70er Jahre war es,

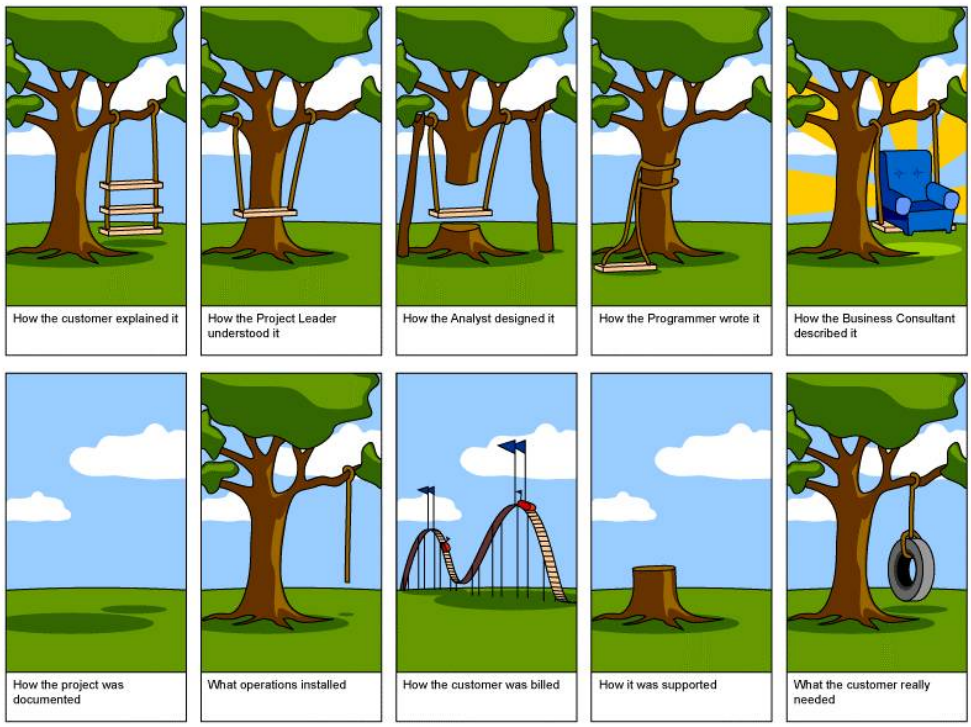


Abbildung 2.1: *Probleme der Produktentwicklung am Beispiel der Entwicklung einer Reifenschaukel, [Zau09].*

einen vollständig algorithmisier- und automatisierbaren Entwicklungsprozess zu erreichen. 1976 wurde in [Fra76] bewiesen, dass es Konstruktionsaufgaben gibt, die nicht algorithmisch lösbar sind. 1985 wurde diese These in [Fra85] wiederholt und durch Praxisbeispiele verdeutlicht. Da sich also zeigte, dass eine vollständige Algorithmisierung und Automatisierung nicht möglich ist, wurden verstärkt Vorgehensmodelle erzeugt, die den Entwickler bei seiner Arbeit unterstützen und ihm einen Leitfaden an die Hand geben sollen. Da für die Vielzahl der unterschiedlichen Entwicklungsprobleme kein allgemeingültiges, flexibel einsetzbares Vorgehen beschrieben werden kann, wurden in den folgenden Jahren eine ganze Reihe zum Teil sehr unterschiedlicher Vorgehen entwickelt.

Vorgehensmodelle lassen sich z.B. nach [Lin05, S. 38] in die Kategorien Mikrologik und Makrologik aufteilen. Eine Mikrologik ist eine Abfolge von elementaren Handlungsabläufen (z.B. TOTE-Modell = Test-Operate-Test-Exit), während eine Makrologik ein Vorgehen auf der Basis von Arbeitsabschnitten oder Phasen beschreibt (z.B. VDI-RL2221 [VDI93]). Dazwischen liegt ein Vorgehen als Abfolge operativer Arbeitsschritte (z.B. Problemlösungszyklus [Ehr07]). Diese klare Einteilung ist jedoch häufig nicht ohne weiteres möglich, da die Mikrologik und Problemlösung oftmals in eine Makrologik eingebettet sind.

Im Rahmen dieser Arbeit wurde eine umfassende Analyse von Produktentwicklungsprozessen und speziellen Vorgehensweisen aus den Bereichen Systementwicklung/ Mechatronik, Maschinenbau, Elektrotechnik und Softwareentwicklung durchgeführt. Dazu wurde die Literatur zu 60 verschiedenen Vorgehensweisen (von 1954 [Kes54] bis heute) untersucht. Für jede Vorgehensweise wurden die Basisaktivitäten (z.B. Zielobjekt benennen) und die Wechselwirkungen zu weiteren Basisaktivitäten zusammengestellt. Um Redundanzen zu vermeiden, wurden die Basisaktivitäten auf einheitliche Begriffe zurückgeführt. Ergebnis der Analyse war die Erkenntnis, dass sich die Produktentwicklung im Sinne einer fachbereichsübergreifenden Makrologik auf die in Abb. 2.2 a) gezeigten allgemeinen Bausteine zurückführen lässt. Außerdem wurden die wesentlichen Aktivitäten identifiziert, die zur Durchführung einer modernen Produktentwicklung notwendig sind. Die Tabelle A.1 im Anhang dieser Arbeit enthält eine Zusammenfassung der unter-

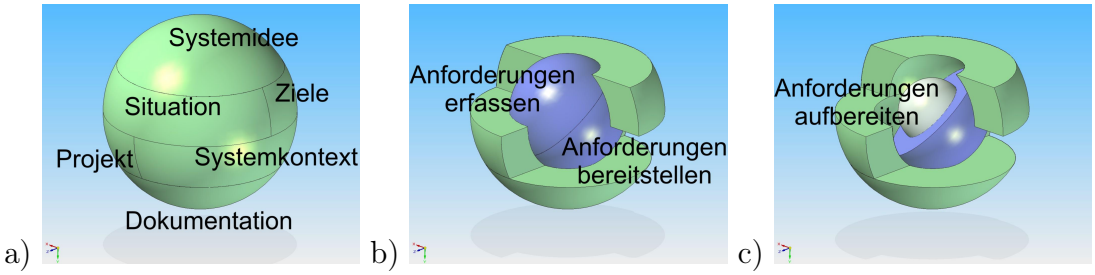


Abbildung 2.2: *a) Allgemeine Bausteine der Produktentwicklung b) Die „äußere“ Anforderungsverarbeitung und c) die „innere“ Anforderungsverarbeitung.*

suchten Vorgehensweisen zur Produktentwicklung. In der Kopfzeile sind die identifizierten allgemeinen Bausteine der Produktentwicklung dargestellt. Für jeden in den Zeilen eingeordneten Ansatz wurde untersucht, inwieweit diese Bausteine berücksichtigt werden. Eine Abstufung gibt an, ob die Bausteine auch methodisch hinreichend unterstützt werden. Die Bausteine und Aktivitäten werden in den folgenden Abschnitten genauer betrachtet und erläutert.

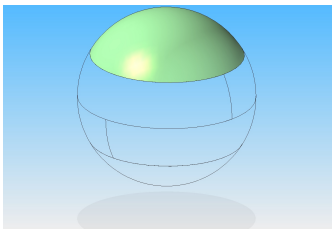
Die Darstellung in Form einer Kugel verdeutlicht, dass es sich bei der Produktentwicklung entgegen einigen Vorgehensweisen nicht um einen sequenziellen Prozess handelt. Zwar sind in vielen Vorgehensweisen Iterationen vorgesehen, der wesentliche Ablauf bleibt allerdings sequenziell und eine wesentliche Bearbeitungsrichtung ist von oben nach unten („top-down"). In aktuellen Veröffentlichungen wird dagegen eine neue Denkweise der holistischen Aufgabenbearbeitung ohne einen allgemeingültig festgelegten Startpunkt oder eine Reihenfolge der Aufgabenabarbeitung vorgestellt [Fra98] [Fra05a] [Mot08].

Den Kern der Produktentwicklung bildet das Requirements Management (RM) mit den Komponenten „Anforderungen erfassen", „Anforderungen aufbereiten" und „Anforderungen bereitstellen" (vgl. Abb. 2.2 b) und c)). Es ist die wesentliche Verbindung zwischen allen Bausteinen und trägt somit maßgeblich zur erfolgreichen Produktentwicklung bei.

Es zeigen sich unterschiedliche Schwerpunkte in den verschiedenen Ansätzen. Einige Punkte wurden bisher jedoch kaum betrachtet, wie z.B. eine integrierte Betrachtung der Team- und Planungsaspekte. Aber auch die Tiefe und Zweckmäßigkeit der betrachteten Aspekte innerhalb der Ansätze muss näher untersucht werden. So ist beispielsweise das Bereitstellen von Anforderungen in den meisten Fällen lediglich durch Lasten-/Pflichtenheft oder Anforderungsliste abgedeckt. Eine integrierte den interdisziplinären Entwicklungsprozess begleitende Modellierung zur Unterstützung der Entwickler zumindest auf abstrakter Ebene fehlt bisher. In den folgenden Kapiteln werden die Analyseergebnisse für die einzelnen in Abb. 2.2 dargestellten Bausteine kurz zusammengefasst.

2.1.1 Systemidee beschreiben

Bereits *Kesselring* [Kes54, S. 229] bezeichnet das Auffinden der Aufgabe als den wesentlichen Schritt des „Erfindungsaktes". Der eXtreme Programming (XP)-Ansatz der Softwareentwicklung nutzt Metaphern, um in wenigen klaren Worten die Kernidee des Systems anzugeben [Bec00], [Wol05, S. 11]. Dadurch soll jedem Teammitglied eine Richtung vorgegeben und so eine konsistente Entwicklung erreicht werden. Im *Volere Requirements Specification Template* [Rob08] soll mit dem ersten Gliederungspunkt „*The Purpose of the Project*" ebenfalls eine Zielsetzung des Projekts angegeben werden. Auch *Weilkiens* sieht für das Systems Engineering (SE) die Definition einer Systemidee als wesentlichen Punkt das Entwicklungsteam dazu zu veranlassen in die gleiche Richtung zu arbeiten [Wei06, S. 37].



- Wesenskern des Systems beschreiben

Abbildung 2.3: *Baustein der Produktentwicklung „Systemidee beschreiben“.*

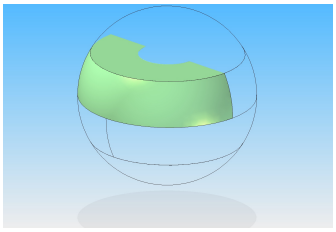
Im Sinne dieser Arbeit soll die Beschreibung der Systemidee in kurzer und knapper Form den Wesenskern des Systems beschreiben (vgl. Abb. 2.3). Hier werden also diejenigen Ziele definiert, die einen übergeordneten Charakter besitzen und prinzipiell durch unterschiedliche Produkte lösbar sind (z.B. transportieren eines Gutes). Sie werden z.B. ausgehend vom Kunden als Bedarf formuliert (vgl. *market-pull*). Oder es wird aus einem vorhandenen Produkt die Systemidee abgeleitet, die nun als Verkaufsargument z.B. dem Marketing zur Verfügung steht (vgl. *technology-push*).

Es gibt im wesentlichen vier Gruppen von Zielen auf Ebene der Systemidee [Krä07, S. 54][Küp93, S. 212f]: Qualitätsziele, Kostenziele, Liquiditätsziele und Sicherungsziele.

2.1.2 Situation analysieren

Die Notwendigkeit, die die Aufgabenstellung umgebende Situation zu analysieren, wird in den meisten der untersuchten Ansätze herausgestellt. Besonders hervorzuheben sind *Franke* [Fra75] [Fra76, S. 136ff], *Birkhofer* [Bir80, S. 15ff], *Ullman* [Ull92, S. 113ff], *Weilkiens* [Wei06, S. 30ff] und *Müller* [Mül08, S. 8].

Franke führt Suchmatrizen zur Anforderungsklärung ein. Diese zeigen bereits Relationen zwischen z.B. dem Produktlebenslauf und der Produktumgebung auf und bilden damit eine sehr gute Grundlage zur systematischen Erfassung von Anforderungen. *Birkhofer* weist darauf hin, dass in allen Lebenslaufphasen aus der Produktumgebung Anforderungen an das Produkt gestellt werden. Dabei kommt der Nutzungsphase eine besondere Bedeutung zu: Hier kann bei bestimmten Produkten ein großer Teil der Umweltbeeinträchtigungen [Obe06, S. 16ff] und Kosten [Krä07, S. 4] entstehen, während die beeinflussenden Parameter bereits in frühen Phasen festgelegt wurden. *Ullman* fordert dazu auf, zunächst die relevanten Kunden und weitere an der Produktentstehung beteiligte Personen zu identifizieren, um diese an der weiteren Produktentwicklung (z.B. in Marktstudien und durch Interviews) zu beteiligen. In *Weilkiens* „Analyse-Vorgehensmodell“ ist jeweils eine Aktivität zur näheren Beschreibung der Produktumgebung (hier Systemkontext) und notwendiger bzw. vorstellbarer Anwendungsfälle vorgesehen. Die Anwendungsfälle sind nur dann vollständig beschrieben, falls die mit ihnen in Beziehung stehenden *Stakeholder* genannt werden. Der Begriff Stakeholder wird



- Stand der Technik / Schwachstellen ermitteln, Benchmark durchführen
- Produktlebenszyklus aufstellen
- Informationsbeschaffung aus Markt, Umwelt und Unternehmen
- Stakeholder identifizieren
- Szenarios beschreiben
- Anwendungsfälle modellieren

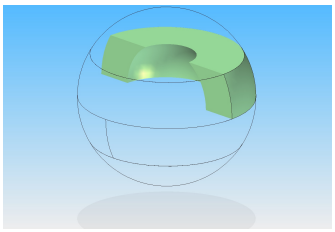
Abbildung 2.4: *Baustein der Produktentwicklung „Situation analysieren“.*

hier für natürliche, juristische und abstrakte Personen bzw. Rollen verwendet, die direkten oder indirekten Einfluss auf Anforderungen haben (vgl. Definition im Glossar). *Müller* beschreibt einen dreistufigen Ansatz für die Entwicklung von Product-Service Systems (PSS). Der erste Prozessschritt ist die Analyse des Systemkontexts. Dieser enthält verschiedene Schichten, die z.B. die verschiedenen Aktivitäten im Produktlebenslauf, die beteiligten Stakeholder und die technische Peripherie berücksichtigen.

Aus allen untersuchten Arbeiten können die in Abb. 2.4 dargestellten Aktivitäten für den Baustein „Situation analysieren“ identifiziert werden. Es wird ein Ist-Zustand erfasst, der den aktuellen Stand der Technik wiedergibt (z.B. durch Benchmarks und anderweitige Informationsbeschaffung) und Schwachstellen bestehender Lösungen (falls vorhanden) identifiziert. Außerdem wird der zu erwartende Produktlebenszyklus des zu entwickelnden Produktes beschrieben. Dazu werden mögliche und gewünschte Anwendungsfälle und ihre in Beziehung stehenden Stakeholder den Lebenslaufphasen zugeordnet. Dabei werden unterschiedliche Szenarien betrachtet (z.B. worst-case und best-case Szenarien). Im Allgemeinen lassen sich sechs Klassen von Szenarien unterscheiden, die hierzu eingesetzt werden können [Ang08, S. 4]. Es handelt sich um explorative Szenarien, derzeitige und zukünftige Anwendungsszenarien, mögliche Problemszenarien, Interaktionsszenarien und validierende Szenarien.

2.1.3 Ziele formulieren

Der nächste betrachtete Baustein im Produktenwicklungsprozess ist das Formulieren von Zielen. Die folgende Definition stammt von *Ehrlenspiel* [Ehr07, S. 89]: „*Ziele sind Soll-Vorstellungen vom Auftraggeber und können unscharf sein. Die Konstruktion formuliert diese Ziele in Anforderungen (Solleigenschaften) um, um sie für sich in einer Anforderungsliste bearbeitbar zu machen.*“ Wesentlicher Schwachpunkt dieser Definition ist die einseitige Fixierung auf die Vorstellungen der Auftraggeber. Ziele sind zum einen Kundenwünsche, die je nach Produktart bereits mehr oder weniger technisch geprägt sind (z.B. „*hohe Nutzlast erzielen*“). Zum anderen sind es Zielsetzungen der beteiligten Unternehmen bzw.



- Zielobjekt benennen
- Zweck benennen
- Zielgruppe und Marktsegment festlegen

Abbildung 2.5: *Baustein der Produktentwicklung „Ziele formulieren“.*

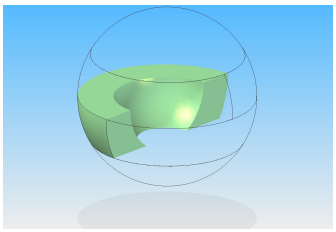
Abteilungen, die erfüllt werden müssen (z.B. „*geringe Herstellkosten erzielen*“). Des weiteren können Ziele aus Randbedingungen, die z.B. aus der Gesetzgebung folgen, entstehen (z.B. „*hohen Umweltstandard erzielen*“). Die Ziele bilden Ausgangspunkte für das Erfassen von Anforderungen. Das geht weit über das reine transformieren von Kundenwünschen in technische Anforderungen hinaus. Anforderungen sind deutlich konkreter als Ziele und beschränken sich nicht nur auf Solleigenschaften des Produktes, sondern können auch z.B. terminliche Vorgaben ausdrücken (vgl. Abschnitt 3.4.1). Darüberhinaus sind mehrere Anforderungen notwendig, um ein Ziel zu erfüllen. Während der Produktentwicklung werden Ziele genutzt, um eine Richtung vorzugeben: Alle Projektbearbeiter sollen „an einem Strang ziehen“.

Dieser Baustein wird von etwa 60% der untersuchten Ansätze gefordert, eine klare Trennung zwischen Zielen und Anforderungen wird in den meisten Fällen jedoch nicht vorgenommen. Lediglich *Franke* [Fra09, S. 4-9], *Lindemann* [Lin05, S. 39ff], *Frank* [Fra06] und *Weilkiens* [Wei06, S. 30f] sprechen explizit von Zielen, die auf einer höheren Abstraktionsebene als Anforderungen anzuordnen sind. *Roth* [Rot94, S. 59] und *Koller* [Kol94, S. 8ff] nutzen die Begriffe Hauptaufgabensatz und Zweck, um den Wesenskern der Aufgabe zu beschreiben.

Im Wesentlichen lassen sich für diesen Baustein die in Abb. 2.5 dargestellten Aktivitäten angeben. Die Ziele müssen einem Zielobjekt zugeordnet werden, welches in der Lage ist den angegebenen Zweck zu erfüllen. Außerdem soll in diesem Zusammenhang eine Zielgruppe (bzw. Marktsegment) festgelegt werden, die das zu entwickelnde Produkt oder Dienstleistung nutzen soll.

2.1.4 Projekt planen und überwachen

Ein weiterer Baustein im Produktentwicklungsprozess ist es, das Projekt zu planen und den Plan zu überwachen. Dieser Baustein wurde schon früh als notwendig erachtet und im Sinne von Zeit- und Terminplanungen umgesetzt. Doch lediglich *Ullman* [Ull92, S. 112] weist explizit darauf hin, dass auch die Teambildung geplant werden muss. Dabei kommt es insbesondere auf die problemspezifische Zusammenstellung der Teammitglieder an. Außerdem muss die Projektplanung fortlaufend überprüft und ggf. überarbeitet werden. Diese Aussagen konnten in praktischen Untersuchungen unter anderem von *Birkhofer* und *Lin-*



- Tätigkeiten ermitteln und planen
- Schwierigkeitsgrad und Entwicklungsrisiko ermitteln
- Zeitbedarf, Personalbedarf und Ressourcenbedarf ermitteln
- Produkt- und Produktentwicklungskosten abschätzen
- Entwicklungsteam formen
- Kommunikationsstrukturen festlegen
- Projekt überwachen

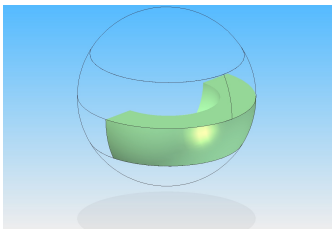
Abbildung 2.6: *Baustein der Produktentwicklung „Projekt planen und überwachen".*

demann [Bir91] [Lin98] bestätigt werden. Im Rahmen einer Studie [Lin98, S. 302] konnten etwa ein Viertel der Probleme im Entwicklungprozess auf eine unzureichende Teamarbeit zurückgeführt werden. Weitere 7% resultierten aus schlechter Arbeitsteilung.

In Abb. 2.6 sind zusammengefasst die Tätigkeiten abgebildet, die für diesen Baustein identifiziert wurden. Dabei handelt es sich in erster Linie um die bekannten Tätigkeiten der Projektplanung: Das Ermitteln und Planen von Tätigkeiten, sowie das Ermitteln und Zuordnen von Schwierigkeitsgrad und ggf. Entwicklungsrisiko. Darauf aufbauend wird der Zeit-, Personal- und Ressourcenbedarf ermittelt und—soweit möglich—die Produkt- und Produktentwicklungskosten abgeschätzt. Sind diese Randbedingungen geklärt, so muss ein geeignetes Projektteam zusammengestellt und geformt werden. Dabei sind neben den rein fachlichen Aspekten ebenso menschliche und methodische Aspekte zu berücksichtigen. Insbesondere müssen geeignete Kommunikationsstrukturen (sowohl zwischenmenschlich als auch in Form von Datenaustauschformaten und -wegen) festgelegt werden. Schlussendlich muss die Einhaltung sämtlicher Pläne auch überwacht und Vereinbarungen eingefordert werden. Dabei ist zum einen der Projektstatus (d.h. Zeit- und Terminziele) und zum anderen die Zielerreichung hinsichtlich der geforderten Produkteigenschaften zu überprüfen. Während der Produktentwicklung müssen also kontinuierlich die Teilsysteme für sich und abschließend das integrierte Gesamtsystem getestet werden.

2.1.5 Systemkontext beschreiben

Das eigentliche Ziel eines jeden Produktentwicklungsprozesses ist es ein Produkt zu entwickeln. Der Erfolg wird am fertigen Produkt und dem benötigten Zeit- und Kostenrahmen gemessen. Innerhalb dieses Bausteins werden für diese Arbeit alle konzeptionellen und gestaltenden Maßnahmen zusammengefasst. Nahezu alle der betrachteten Vorgehensweisen bieten ein geeignetes—z.T. methodisches—Vorgehen an. Dieses ist oftmals an die speziellen Gegebenheiten des betrachteten Fachbereichs angepasst. Die Ansätze, die den Systemkontext nicht betrachten, befassen sich allgemein mit der Planung ohne dabei jeweils in die Tiefe zu gehen (z.B. [VDI80]) oder speziell mit der Aufgabenklärung innerhalb eines etablierten



- Produkt segmentieren / Teilaufgaben definieren
- Ein- und Ausgangsgrößen bestimmen
- Spezifischen Entwurf durchführen
- System integrieren
- System testen

Abbildung 2.7: *Baustein der Produktentwicklung „Systemkontext beschreiben“.*

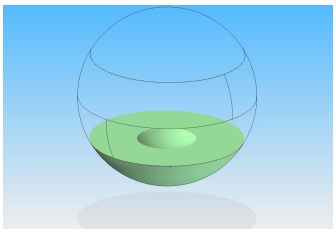
Vorgehens (z.B. [Kru00]).

In Abb. 2.7 sind einige übergeordnete Tätigkeiten aufgelistet, die innerhalb dieses Bausteins durchzuführen sind und in den meisten Vorgehensweisen in dieser oder ähnlicher Form beschrieben werden (z.B. [VDI04]). Das Produkt muss segmentiert bzw. es müssen Teilaufgaben definiert werden, so dass die Entwicklung innerhalb kleinerer und von den einzelnen Entwicklern überschaubarer Segmente durchgeführt werden kann. Für jede Teilaufgabe müssen die Ein- und Ausgangsgrößen bestimmt werden, die die Schnittstellen zwischen verschiedenen Entwicklungsteams darstellen. Innerhalb einer Teilaufgabe bzw. eines Produktsegments findet der spezifische Entwurf statt. Hier werden fachbereichsspezifische Methoden und Werkzeuge genutzt. Beispielsweise kann die kinematische Auslegung einer parallelkinematischen Roboterstruktur durch die Berechnung der Jacobi-Matrix in einer Mathematiksoftware (z.B. Matlab) durchgeführt werden. Die Auslegung der Steuerung erfolgt in einer regelungstechnischen Simulationsumgebung (z.B. Simulink als Zusatzprodukt zu Matlab). Außerdem werden in vielen Teilaufgaben die bekannten—und hier nicht näher zu erläuternden—systematisierenden (z.B. Funktionsstrukturen) und kreativen (z.B. Brainstorming, Synektik) Methoden verwendet. Schließlich müssen die Teillösungen ggf. stufenweise zu einem Gesamtsystem integriert werden.

2.1.6 Fachwissen und Entscheidungen dokumentieren

Die saubere Dokumentation des Entwicklungsprozesses wurde bereits von *Kesselring* in der Mitte des vergangenen Jahrhunderts gefordert. Die Erkenntnis, dass eine Dokumentation der erarbeiteten Lösungen sinnvoll ist, hat sich etabliert und es wurden umfangreiche Methoden entwickelt um Wissen zu speichern und zu handhaben. Dabei sind insbesondere Konstruktionskataloge zu nennen (vgl. [Rot71], [Rot72], [Rot94], [Kol94]), die im Laufe der Jahre durch Rechnerunterstützung an Anwendbarkeit und Zugriffsmöglichkeiten gewonnen haben (z.B. [Fra04] [Kir09]).

Trotz aller Weiterentwicklungen auf den Gebieten der Methodik und der Datenhaltung wird in keinem der untersuchten Ansätze eine zufriedenstellende Dokumentation gefordert. Zwar sollen in jedem Ansatz die Ergebnisse dokumentiert werden, aber nur selten wird die Schaffung eines gemeinsamen Sprachgebrauchs—



- Daten erfassen
- Daten verwalten
- Daten bereitstellen

Abbildung 2.8: *Baustein der Produktentwicklung „Fachwissen und Entscheidungen dokumentieren“.*

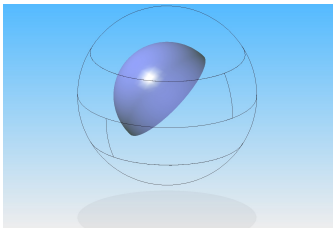
einer Terminologie—als dokumentierter Fachwortschatz (vgl. [Fra02a], [Wei06]) für das Entwicklungsteam gefordert. Außerdem werden keine Entscheidungswege dokumentiert, die das Projekt auch für Neueinsteiger transparent machen und verhindern, dass Fehler und nicht zielführende Entwicklungspfade mehrfach begangen werden.

Abb. 2.8 symbolisiert den Baustein der Dokumentation. Dieser muss das Erfassen, Verwalten und Bereitstellen von Fachwissen und getroffenen Entscheidungen beinhalten.

2.1.7 Anforderungen erfassen

Das Erfassen von Anforderungen wird bis auf wenige Ausnahmen in allen Ansätzen explizit berücksichtigt. Viele Ansätze bieten für eine möglichst vollständige und aussagekräftige Erfassung von Anforderungen eine umfangreiche Methodik und ein breites Spektrum von Hilfsmitteln an (z.B. [Fra76] [Löh91] [Ull92] [Rot94] [Dae02] [Lin05] [Fra06] [Wei06] [Ehr07] [Pah07] [Rup07] [Rob08]).

In Abb. 2.9 sind die wesentlichen Tätigkeiten innerhalb dieses Bausteins dargestellt. Ein häufig vernachlässigter Schritt ist die Planung der Anforderungserhebung, die z.B. von *Humpert* [Hum95] besonders gefordert wird. Hier müssen die Randbedingungen festgelegt werden, beispielsweise die sinnvoll zu befragenden Stakeholder und die Informationsmenge, die beim Interview eingebracht wird. Werden z.B. Kunden konkret nach Auffälligkeiten an einem Vorgängerprodukt gefragt (große eingebrachte Informationsmenge), werden die erfassten Anforderungen für die Entwicklung einer neuen (optimierten) Variante sehr hilfreich für



- Anforderungserhebung planen
- Kundenwünsche in technische Anforderungen übersetzen
- Restriktionen und Bedingungen ermitteln
- Anforderungseigenschaften erfassen

Abbildung 2.9: *Baustein der Produktentwicklung „Anforderungen erfassen“.*

eine Neuentwicklung aber teilweise unbrauchbar bzw. zu sehr lösungsbestimmend sein.

Der nächste Schritt ist es, die Kundenwünsche und technischen Anforderungen zu ermitteln. Dabei kann die Gruppe der Kunden sehr weit aufgefasst werden. Teilweise werden unter Kunden alle im Lebenszyklus des zu entwickelnden Produktes involvierten Stakeholder verstanden, d.h. neben dem Betreiber auch Personen der Fertigung oder Entsorgung. Hier bietet es sich an, strukturierte Interviews mit den identifizierten Stakeholdern zu führen und die Anforderungssätze dann z.B. mit Hilfe von Checklisten, Suchwürfeln o.ä. zu ergänzen.

Ein weiterer Punkt ist das Ermitteln von Restriktionen und Bedingungen. Restriktionen ergeben sich häufig aus geometrischen Randbedingungen, z.B. zur Verfügung stehende Bauräume oder Schnittstellen zu bereits vorhandenen Nachbarsystemen. Hierzu muss die Produktumgebung gezielt untersucht werden. Bedingungen ergeben sich z.B. daraus, dass eine Anforderung nicht immer (d.h. zu jeder Zeit) oder überall (d.h. an jedem Ort) erfüllt bzw. gleich erfüllt sein muss. Beispielsweise muss ein Fahrzeug bei sehr hohen Geschwindigkeiten (z.B. Autobahnfahrt) genug aerodynamischen Abtrieb für eine sichere Handhabung erzeugen. Bei vergleichsweise niedrigen Geschwindigkeiten (z.B. Stadtverkehr) spielt der Abtrieb hingegen keine Rolle.

Schließlich müssen für alle Anforderungen die Anforderungseigenschaften ermittelt werden. Dabei handelt es sich um ihre Priorisierung (z.B. nach Wünschen, Fest- und Mindestforderungen) und die Festlegung von Werten, z.B. *Das Fahrzeug soll bei Autobahnfahrten eine Maximalgeschwindigkeit von 195 km/h erreichen.*

2.1.8 Anforderungen bereitstellen

Das Bereitstellen der erfassten Anforderungen wird ebenfalls in den meisten untersuchten Ansätzen verlangt. Es werden strukturierte Anforderungslisten, Spezifikationen als Entwicklungsgrundlage oder Lasten- und Pflichtenheften als Vertragsbestandteile gefordert. Insbesondere bei Auftragsentwicklungen ist ein rechtlich bindendes Vertragsdokument notwendig, welches dem Kunden die Sicherheit gibt nur für etwas bezahlen zu müssen, was er verlangt hat bzw. Nachbesserungen für nicht zufriedenstellende Lösungen einzufordern. Auf der anderen Seite gibt es dem Entwickler die Sicherheit, dass er bei Erfüllung der festgelegten Kriterien für seine Arbeit bezahlt wird. Allerdings wird auch—insbesondere in Ansätzen der IT —festgestellt, dass Anforderungen sich im Verlauf der Produktentwicklung ändern können. Wenn erste Prototypen erstellt wurden zeigt sich insbesondere bei schwer quantifizierbaren Anforderungen, ob sie tatsächlich so gemeint waren, wie ursprünglich festgelegt.

Besonders hervorzuheben sind hier die Ansätze von *Humpert*, *Kruse*, *Weilkiens* und *Rupp*. *Humpert* entwickelt in seiner Dissertation [Hum95] ein objektorientiertes Anforderungsmodell. Hier sind Anforderungen als Objekte enthalten und werden in Listen in Diagrammen dargestellt. Im Fall der Diagramme sind Abhän-

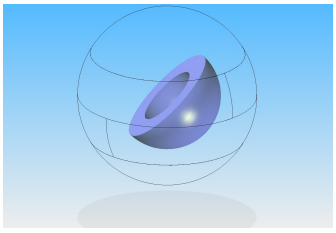
gigkeiten deutlich und intuitiv erkennbar. Ähnliche Darstellungen beschreibt auch *Weilkiens* [Wei06]. Er nutzt hierfür die objektorientierte Notation der SysML, die bereits die Stereotypen `<<requirement>>` und einige standardisierte Beziehungstypen bereitstellt (vgl. Abschnitt 3.3.1). In speziellen Diagrammen wird ebenfalls eine bestimmte Sicht auf das Anforderungsmodell visualisiert.

Kruse führt Kennzahlen ein, die einem Projektleiter die Beurteilung der Anforderungsqualität erleichtern sollen [Kru96, S. 92]. *Rupp* führt ebenfalls umfangreiche Kennzahlen (hier als Metriken bezeichnet) für die Beurteilung des Anforderungsmodells ein [Rup07, S. 360] [Pik06]. Darauf wird in Abschnitt 3.4.3 näher eingegangen.

Rupp und *Weilkiens* weisen darauf hin, dass im Informationstechnologie (IT)-Bereich nicht alle Anforderungen sinnvoll in rein textueller Form abbildbar sind. Oftmals ist es einfacher z.B. Reihenfolgen von zu implementierenden Tätigkeiten in formalisierten Anwendungsfällen oder z.B. in Sequenzdiagrammen darzustellen.

Im Bereich des (Industrie)Designs arbeiten Designer mit sogenannten „*Moodboards*“: Eine Zusammenstellung von Bildern, Fotos und kurzen Texten oder Schlagworten, die in ihrer Gesamtheit eine bestimmte Stimmung (*mood*) ausdrücken. Die geeignete Art der Repräsentation einer Anforderung hängt also sehr stark vom Hintergrund derjenigen Person ab, der die Anforderungen bereitgestellt werden sollen und welche Information transportiert werden soll. *Bauer* stellt in [Ste08a] eine Darstellungsmöglichkeit vor, die den unterstützenden bzw. konfliktären Zusammenhang von Anforderungen in einem dreidimensionalen Raum visualisiert. Eine solche Darstellungsmöglichkeit ist geeignet, um einfache Zusammenhänge mit technisch und methodisch wenig vorgebildeten Personen zu besprechen.

Die in diesem Baustein durchzuführenden Tätigkeiten sind in Abb. 2.10 gezeigt. Es müssen geeignete, voneinander verschiedene Repräsentationen der Anforderungen erstellt werden, die als juristisch gültige Vertragsbestandteile (als Abnahmekriterien) verwendet werden. Zudem sollen sie als Hilfsmittel für unterschiedliche Entwicklungsprozesse (z.B. in den verschiedenen Fachbereichen) geeignet sein. Außerdem soll ein „Anforderungsverfolgungsüberblick“ erstellt werden, der es erlaubt Änderungen und deren Auswirkungen nachzuvollziehen. Schließlich



- Anforderungsliste, Lasten- und Pflichtenheft erstellen
- Anforderungsverfolgungsüberblick erstellen
- Modellkennzahlen erstellen

Abbildung 2.10: *Baustein der Produktentwicklung „Anforderungen bereitstellen“.*

müssen für eine Projektüberwachung geeignete Kennzahlen bereitgestellt werden.

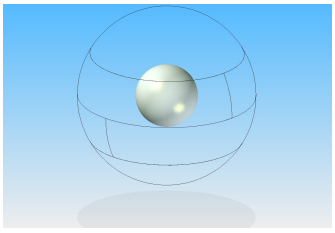
2.1.9 Anforderungen verarbeiten

Die Verarbeitung der Anforderungen ist als der innere Kern einer erfolgreichen Produktentwicklung zu betrachten (vergleichbar dem Keim eines Aprikosenkerns). Er bildet das verbindende Glied zwischen der (oft wenig strukturierten, unklaren und Redundanz behafteten) Erfassung von Anforderungen und der geeigneten Bereitstellung der Informationen. Die Notwendigkeit die erfassten Anforderungen vor der Bereitstellung zu verarbeiten wird in deutlich weniger der betrachteten Ansätze thematisiert als es noch für die Erfassung und Bereitstellung der Fall war. Allerdings ist das Thema noch immer in mehr als der Hälfte der Ansätze zu finden.

Die Mehrzahl der genannten Maßnahmen beschränken sich darauf, Kundenanforderungen in technische Anforderungen zu übersetzen, die Anforderungen zu gliedern und zu klassifizieren, Redundanzen und Inkonsistenzen zu finden und ggf. zu beseitigen. Teilweise wird darauf eingegangen, Beziehungen zwischen Anforderungen und weiteren Anforderungen [Kru00], Systemkomponenten [Kru96] und Anwendungsfällen [Wei06] zu etablieren, um diese für die Produktentwicklung und Modellauswertung (z.B. Identifikation von Zielkonflikten) zu nutzen (vgl. Abschnitt 3.4). *Humpert* [Hum95] und *Jung* [Jun06] geben an, dass die Beziehungen darüber hinaus genutzt werden können, um neue Anforderungen zu finden und so das Anforderungsmodell zu vervollständigen.

Die grundlegend notwendigen Tätigkeiten in diesem Arbeitsschritt sind (vgl. Abb. 2.11) das Aufstellen einer Anforderungsklassifikation und Festlegung der Anforderungseigenschaften. Dafür müssen auch die (z.B. natürlich sprachlich erfassten) Kundenwünsche in technische Anforderungen übersetzt werden. Es wird eine Anforderungsstruktur erstellt, so dass das Anforderungsmodell analysiert, umstrukturiert, bereinigt und bewertet werden kann. Abschließend muss das Anforderungsmodell von einer verantwortlichen Instanz verabschiedet werden.

In der Literatur (z.B. [Fra76] [IEE98a] [IEE98b] [Sch02] [Rup07]) finden sich eine Reihe von Eigenschaften, die das freigegebene Anforderungskollektiv aufweisen soll. Diese sind im Anhang (Abschnitt A.2) beschrieben.



- Anforderungsklassifikation aufstellen
- Anforderungseigenschaften aufstellen
- Kundenwünsche in technische Anforderungen übersetzen
- Anforderungsstruktur erstellen
- Anforderungsmodell analysieren, umstrukturieren, bereinigen und bewerten
- Anforderungsmodell verabschieden

Abbildung 2.11: *Baustein der Produktentwicklung „Anforderungen verarbeiten“.*

Ob eine Anforderung oder ein Anforderungskollektiv die oben beschriebenen Eigenschaften aufweist, ist im Einzelfall schwer zu entscheiden. Beispielsweise könnte die Vollständigkeit eines vorliegenden Anforderungskollektivs nur dann beurteilt werden, falls ein vollständiges Anforderungskollektiv zum Vergleich bekannt wäre. Das ist nie der Fall. Natürlich kann eine sehr aufwändige Anforderungsklärung zu einer sehr umfangreichen Anforderungssammlung führen, die dann mit hoher Wahrscheinlichkeit nahezu vollständig, allerdings nicht minimal wäre.

Widersprüche zwischen Anforderungen entstehen oftmals erst durch die Wahl von Prinzipen oder die Ausgestaltung, d.h. die Entscheidung über die Konsistenz eines Anforderungskollektivs ist a-priori kaum möglich. Eine vollständige Unabhängigkeit, wie z.B. vom Unabhängigkeitsaxiom des „*axiomatic design*“ [Suh90] gefordert, ist eine theoretische Hypothese aber praktisch unrealistisch.

Innerhalb des Anforderungskollektivs soll jede einzelne Anforderung bestimmte Eigenschaften aufweisen. Diese sind aus unterschiedlichen Literaturquellen entnommen (z.B. [Bid68] [Kic95] [IEE98b] [Sch02] [Rup07] [Tav08]) und im Anhang (Abschnitt A.2) beschrieben.

Kickermann beschreibt in seiner Dissertation ausführlich eine einfache Anforderungssyntax, die zu eindeutigen, unmissverständlichen und konsistenten Anforderungsbezeichnungen führt [Kic95, S. 85ff]. Insbesondere eignet sich die dort beschriebene Syntax für eine rechnerunterstützte Weiterverarbeitung durch eine EXPRESS/STEP-konforme Beschreibung.

2.2 Verteilte Produktentwicklung

Die Entwicklung komplexer Produkte findet immer häufiger in Form einer interdisziplinären Zusammenarbeit in Kooperationsnetzwerken statt. Produktentwicklung kann demnach plakativ als „Teamsport“ bezeichnet werden. Abb. 2.12

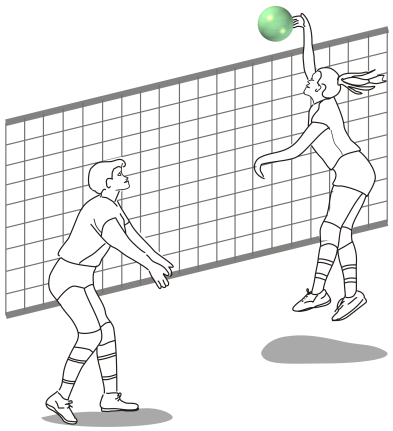


Abbildung 2.12: *Die verteilte Produktentwicklung als Teamsport.*

stellt ein Match mit der Produktentwicklung als Volleyball dar, der von einem Spieler zum anderen bewegt wird. Wie beim Volleyball gibt es bestimmte Techniken und Strategien, die für bestimmte Randbedingungen (z.B. Feldposition) eingesetzt werden können. Das Gesamtziel ist es immer, das Spiel zu gewinnen. Dabei ist jeder Spieler wichtig, egal ob er eine Vorlage gibt oder einen Punkt erzielt. Keine Position ist nur Dienstleister für eine andere. Genauso werden nur im frühen gleichberechtigten Zusammenspiel der unterschiedlichen Fachdisziplinen Synergieeffekte der Produktentwicklung effizient genutzt sowie Fehler und unnötige Iterationen vermieden. Allerdings besteht auch die Gefahr durch zu viele geschlossene Kompromisse keine wirklich innovativen Produkte zu entwickeln.

Die verteilte Produktentwicklung stellt eine typische Arbeitsbedingung dar, welche komplexe Anforderungsstrukturen hervorbringt. In dieser Arbeit wird unter einem Kooperationsnetzwerk eine Gruppe organisatorisch und örtlich getrennter Unternehmen oder Abteilungen verstanden, die gemeinsam an der Entwicklung eines bestimmten Produktes arbeiten. Dabei ist das Ziel die spezielle Erfahrung und das spezifische Wissen der verschiedenen Unternehmen aus unterschiedlichen Fachbereichen zur gemeinsamen Produktentwicklung zu nutzen [Fra05b], z.B. zur Entwicklung mechatronischer Produkte wie Roboter, Kraftfahrzeuge oder Flugzeuge. Dieses Vorgehen berücksichtigt, dass insbesondere kleine Unternehmen nicht in vielen unterschiedlichen Bereichen großes Wissen und Erfahrung besitzen, aber dennoch Experten und Weltmarktführer in einer Nische sein können.

Außerdem werden durch die Parallelisierung von Arbeitspaketen in Kooperationsnetzwerken die Kosten und die Markteinführungszeit gesenkt und gleichzeitig die Qualität gesteigert. Durch das *Simultaneous Engineering* bzw. *Concurrent Engineering* sollen laut [Ste93] 30-60% geringere Kosten, 30-90% kürzere Entwicklungszeiten und 30-87% verbesserte Qualität erzielt werden.

Abgesehen von diesen Vorteilen bringt die Entwicklung in Kooperationsnetzwerken auch einige Schwierigkeiten und Gefahren mit sich. Die wichtigsten Aspekte wurden aus der einschlägigen Literatur herausgearbeitet und nach Themenfeldern gegliedert systematisch zusammengestellt. In den folgenden Abschnitten sind die identifizierten Themenfelder näher beschrieben (vgl. auch [Ste09b]).

2.2.1 Kommunikation zwischen Entwicklungspartnern

Je größer die Entfernung zwischen den Standorten der Entwicklungspartner ist, desto höher ist der Aufwand für die Kommunikation [Gau01]. Es müssen z.B. Dienstreisen unternommen werden oder aufwendige Kommunikationstechnologien (z.B. Videokonferenz, Shared Whiteboard, Application Sharing) angeschafft, administriert und der Umgang mit ihnen erlernt werden. Wurde kein klarer Verhaltenskodex zur Kommunikation definiert, kommt es zu Effizienzverlusten [Gau01]. Verschiedene gesprochene Sprachen und verschiedene kulturelle Hintergründe verstärken diesen Effekt [Ben01]. Durch die Globalisierung kommt es

immer mehr zu einer „Entfremdung“ der Stakeholder und Systementwickler. Das Denken in Abteilungen und Kostenstellen führt zu Verständnisproblemen und Rechtfertigungskämpfen. Zusätzlich entstehen Vertragsgrenzen durch „Outsourcing“ von Entwicklung [Rup07, S. 482]. Externe Entwicklungsabteilungen werden als externe Partner betrachtet und erhalten keinen vollständigen Zugang zu internen Informationen.

Es ist also wichtig, eine gemeinsame Sprache und gemeinsame Vokabeln zu definieren (z.B. ein Glossar zu erstellen, vgl. Abschnitt 2.1.6), um eine effiziente Kommunikation zu gewährleisten. Dieses ist zumindest auf den Abstraktionsebenen notwendig, auf denen ein Austausch der verschiedenen Entwicklungspartner erfolgt, d.h. vornehmlich auf der Systemebene. Im spezifischen Entwurf können hingegen Fachvokabeln benutzt werden, die nicht gemeinsam festgelegt wurden, sofern sie nicht gemeinsam benötigt werden. Dabei muss unterschieden werden, ob die beteiligten Disziplinen aus rein technischen Bereichen kommen oder ob zusätzlich nicht-technische Bereiche involviert sind, die andere Denk- und Sprachmuster verwenden. Außerdem spielt die Qualifikation der beteiligten Entwickler eine große Rolle. Erfahrene Entwickler können aus vergangenen Erlebnissen Querverweise bilden, die es ihnen ermöglichen, sich neue Zusammenhänge zu erschließen. Unerfahrene Entwickler (z.B. Hochschulabsolventen) können sich Begriffe, die sie nicht kennen, häufig nur schwer herleiten (vgl. z.B. [Jän07, S. 131]).

2.2.2 Austausch von Informationen

Zanker gibt den Austausch von Informationen als Hauptschwachpunkt bei der Zusammenarbeit an [Zan99]. Dazu zählen das Speichern, Auswählen und Weiterverarbeiten von Informationen sowie das Wissensmanagement. Nicht immer ist jedem Projektmitglied bekannt, ob es mit einer aktuellen Version der Daten arbeitet. Außerdem müssen Dateien häufig von einem Datenformat in ein anderes konvertiert werden (z.B. unterschiedliche CAD-Systeme), was im Allgemeinen mit einem Informationsverlust einhergeht oder Dateien können aufgrund von nicht verfügbarer Software nicht von allen Projektmitgliedern geöffnet werden [Gau01]. Außerdem werden in unterschiedlichen Fachdisziplinen unterschiedliche Modelle verwendet, die mit unterschiedlichen Werkzeugen erstellt werden. Diese Daten sind häufig nur händisch übertragbar.

Schnittstellen sind oft schlecht definiert und Regeln für den Austausch von Informationen werden nur selten festgelegt [Ste93] [Gau01]. Falls Regeln festgelegt wurden, so führt dieses häufig zu einem größeren Zeitaufwand, um spezielle Notationen und projektspezifische Vorgehensweisen zu erlernen.

Es existieren einige produktspezifische Ansätze (z.B. im Bereich Parallelroboter [Fra05d]), um einen „Projektdatenkoordinator“ bei einem konsistenten Austausch von Daten zu unterstützen und den Austausch von Modelldaten zeiteffizienter zu gestalten. Allerdings müssen solche Entwicklungsumgebungen ständig an neue Technologien angepasst werden. Schon ein neues Release einer CAD-

Software kann die bisherigen Schnittstellen unbrauchbar machen. *Brey* gibt in [Bre03, S. 129] als wesentliche Schwachstellen von Constraint-based development environment (CBDE)-Systemen die Schnittstellenproblematik bei der Solveranbindung und den Anpassungsaufwand auf Grund von Syntaxänderungen bei Versionswechseln an.

Ein Modell des zu entwickelnden Produkts ist auch immer eine Dokumentation der Entwicklungsergebnisse. Es dokumentiert den aktuellen Entwicklungsstand und erlaubt es darüber zu diskutieren. Somit bildet es eine Basis für Repräsentationen des aktuellen Wissens für die beteiligten Stakeholder. Außerdem wird bei Personalwechseln ein neuer Mitarbeiter sich innerhalb kürzerer Zeit in das Thema einarbeiten und in der Lage sein die bisher getätigten Entscheidungen nachzuvollziehen. Darüberhinaus ist es möglich Modelle von Vorgängerprodukten schnell hinsichtlich aktueller Fragestellungen zu analysieren.

Eine erprobte Möglichkeit, um strukturierte Informationen zwischen Entwicklungspartnern auszutauschen sind rechnerunterstützte, internetbasierte Konstruktionskataloge [Fra04]. Die Informationen müssen immer aktuell und für jeden Projektmitarbeiter verfügbar sein. Ist der Mitarbeiter nicht in der Lage von seinem Arbeitsplatz aus in kurzer Zeit die gewünschte Information zu erhalten oder hat er das Gefühl die Datenbank wäre nicht aktuell, so wird die Akzeptanz schnell abnehmen. Daher muss ein Katalog ständig durch neue Informationen erweiterbar sein und der Zugriff muss von jedem Entwicklungsstandort aus möglich sein, egal welche Rechnerplattform und Konfiguration verwendet wird. Ein solches System wurde z.B. für die Entwicklung von Parallelrobotern erstellt [Fra01]. Aktuelle Arbeiten beschäftigen sich damit, diese Systeme derart zu erweitern, dass ein gezielter Zugriff z.B. auf Auslegungsberechnungen möglich ist [Büt07] [Str08] [Kir09].

2.2.3 Arbeitsteilung

Eine unzureichende Klärung der Projektziele führt häufig zu ineffizienter Projektarbeit [Ben01]. Projektziele müssen für jedes Projektmitglied verständlich ausgedrückt vorliegen und allen bekannt sein (vgl. Abschnitt 2.1.3). Die Bearbeitung jeder Teilaufgabe muss einen Beitrag zur Erfüllung der Projektziele liefern.

Eine unzweckmäßige Aufteilung von Teilaufgaben verursacht zusätzliche Arbeit [Ben01]. Außerdem werden Verantwortlichkeiten für Ergebnisse häufig nicht eindeutig festgelegt und nicht kontrolliert [Zan99]. Ein einzelnes Mitglied des Projektteams ist nicht in der Lage das Gesamtsystem zu überblicken. Daher ist es sich möglicher Schwachstellen nicht bewusst und kann die Wichtigkeit einer Schwachstelle in seinem Bereich für das Gesamtsystem nicht abschätzen [Gau01]. Wenn die Aufteilung in Teilprojekte einmal erfolgt ist, so ist es häufig schwierig, diese Aufteilung im Nachhinein wieder zu verändern. Der organisatorische Aufwand für die Koordination der Arbeiten und die Wartung der Schnittstellen ist vergleichsweise hoch [Zan99].

2.2.4 Kombination und Integration von Ergebnissen

In vielen Projekten ist kein kontinuierlicher, systematischer und die Fachbereiche überspannender Ablaufplan vorhanden, so dass die richtigen Ergebnisse nicht zum richtigen Zeitpunkt verfügbar sind (vgl. Abschnitt 2.1.4). Schnittstellen sind oft unzweckmäßig in Art (z.B. Telefonat zur Klärung konkreter konstruktiver Probleme) und Anzahl (z.B. ein Projekttreffen pro Jahr) [Zan99]. So werden häufig „optimale“ Teillösungen erstellt (optimal in Bezug auf das initiale Anforderungskollektiv), die integriert aber ein „suboptimales“ Gesamtsystem bilden. Die Änderungen und das Anwachsen der Anforderungen bleiben häufig unberücksichtigt.

Die Dokumentation von Ergebnissen wird meist als eine lästige Pflicht angesehen. Wenn überhaupt, dokumentieren die Projektmitglieder ihre Ergebnisse zum Projektende oder in einer Form, die nur dem Ersteller selbst verständlich ist. Im Falle von Änderungen der Personalzusammensetzung oder anderen Randbedingungen können Entscheidungen nicht zurückverfolgt werden und im ungünstigsten Fall müssen einige Arbeiten erneut erledigt werden (vgl. Abschnitt 2.1.6).

2.2.5 Menschliches Verhalten und ihre Beziehungen

Eine verbesserte Kommunikation zwischen allen Beteiligten ist eine wichtige Voraussetzung, Fehler zu vermeiden. Bei der verteilten Entwicklung ist der räumliche Abstand zwischen den Entwicklern für eine verringerte Kommunikationswahrscheinlichkeit verantwortlich. Neuere Technologien versuchen den räumlichen Abstand virtuell zu überbrücken (z.B. Instant Messaging, Videokonferenzen, Application Sharing). Häufig sind Projektmitglieder aber zu unerfahren in der Benutzung dieser Technologien oder zu sehr in traditionelle Muster eingefahren, so dass die Akzeptanz vergleichsweise gering ausfällt. Hinzu kommt, dass Entwickler sich selbst häufig als kreative Erfinder sehen, was zu (oft unbewussten) inneren Widerständen bei der Benutzung von (Telekooperations-)Methoden führt [Gau01] [Ben01].

Abgesehen von Effizienzverlusten auf Grund unterschiedlicher Kulturen und Strukturen in den verschiedenen Unternehmen, verfolgt jeder Entwicklungspartner unterschiedliche Ziele und Strategien [Gau01] [Zan99]. Falls Entwicklungspartner sich nicht vertrauen, wird das zu verstärktem Abteilungsdenken der Projektmitglieder führen [Gau01] [Ben01] [Zan99]. Im Allgemeinen folgt daraus das Zurückhalten von Informationen sowie insgesamt verringerte und verschlechterte Kommunikation.

Ehrlenspiel sieht den Menschen als Problemlöser und gibt individuelle (den einzelnen Entwickler betreffende) und äußere Einflüsse an, die aus einer Aufgabe ein Problem machen [Ehr07, S. 52]. Als individuelle Einflüsse werden z.B. Denk- und Handlungsstile sowie Emotionen und Motivation genannt. Äußere Einflüsse sind beispielsweise die verfügbaren Informationen oder die soziale und organisa-

torische Einbindung des Entwicklers.

Ein wichtiges Hilfsmittel für eine erfolgreiche Zusammenarbeit von Menschen ist die graphische Darstellung. Denn sie „*zwingt dazu, die eigene Meinung über Problemfelder, deren Elemente und Zusammenhänge sichtbar und damit diskutierbar zu machen. Es kann dafür oder dagegen argumentiert werden, die Suche nach Fakten wird angeregt*“ [Dae02, S. 128]. Auch aus der pädagogischen Psychologie ist bekannt, dass durch grafische Unterstützung komplexe Zusammenhänge effektiver gelernt werden als rein textbasiert bereitgestellte Information [May90] [Wei91]. Insbesondere kreative Prozesse werden durch bildliche Darstellungen auf allen Abstraktionsstufen verbessert [Lip00]. *Salustri* behauptet in [Sal08], dass obwohl Ingenieure im Allgemeinen als „visuelle Denker“ betrachtet werden, graphische Darstellungen von qualitativen Informationen in frühen Phasen der Entwicklung kaum genutzt werden. Es werden zwei Prinzipien bezüglich des Nutzens diagrammatischer Darstellungen angegeben, die hier wegen ihrer Einprägsamkeit im Original wiedergegeben werden:

- „*Simplicity is power.*
- *Diagrams augment cognition.*“

Bei *Rupp* wird die Beeinflussung der Systementwicklung durch menschliche Einflussfaktoren besonders in der Phase der Systemanalyse hervorgehoben [Rup07, S. 73]. Als Einflussfaktoren werden hier Motivation, kommunikative Fähigkeiten, Art des Wissens, Homogenität der Stakeholdermeinungen, Abstraktionsvermögen, Machtsituation und Gruppendynamik, sowie Kultur und Kompetenz der Stakeholder angegeben. Personen mit großem Fachwissen sind demnach häufig kaum in der Lage ihr Wissen sprachlich zu kommunizieren. Ein Ausweichen auf Kommunikationsmittel, die nicht rein sprachlich funktionieren wird empfohlen. Die Inhomogenität der Stakeholdermeinung führt dazu, dass Anforderungen nicht eindeutig gestellt werden können bzw. Zielkonflikte auftreten. Hier muss ein Konsens oder Kompromiss zwischen den verschiedenen Meinungen gefunden werden.

Weiterhin besteht das Problem der Transformation von Informationen, d.h. der Frage, was der Anforderungssteller wirklich meinte als er die Anforderung formulierte [Rup07, S. 140ff]. Der Mensch bildet aufgrund seines Vorwissens und seiner Erfahrung ein Modell der Realität als „persönliche Wirklichkeit“ aus. Dadurch werden Informationen verfälscht oder gehen verloren. Man spricht von den Darstellungstransformationen Tilgung, Generalisierung und Verzerrung [Ban94]. Der Informationsaustausch über Bilder birgt die Gefahr zu viel Information zu transportieren, d.h. einen ungewollten Interpretationsspielraum zu geben [Rup07, S. 173f].

Ein weiterer menschlicher Einflussfaktor sind Widerstände der Mitarbeiter gegenüber Veränderungsprozessen [Rup07, S. 527]. Menschen begegnen Veränderungen häufig (bewusst oder unbewusst) durch Widerstand. Dieser ist umso größer, je geringer die Möglichkeit der Einflussnahme im Vorfeld war. Im Extremfall

entstehen Widerstände nur deshalb, weil die Mitarbeiter nicht am Veränderungsprozess beteiligt, sondern vor vollendete Tatsachen gestellt wurden.

Widerstand kann sich in verschiedener Weise manifestieren [Rup07, S. 516]. Das Projekt wird verzögert und die Schuld dafür der Veränderung gegeben. Entscheidungen wird zugestimmt ohne sie zu hinterfragen („Dienst nach Vorschrift“). Es werden viele kleine (an sich unproblematische) Fehler als Beweis für die Unzulänglichkeit der Veränderung gefunden. Andere Ziele werden so aufgewertet, dass die Veränderung relativ gesehen einen geringeren Wert erhält. Informationen werden falsch interpretiert oder es werden andere Absichten unterstellt. Einseitige oder falsche Informationsverteilung sorgt für Ablehnung der Veränderung.

2.2.6 Methoden und Werkzeuge

In der multidisziplinären Entwicklung von komplexen Produkten wird eine große Bandbreite verschiedener Methoden und Werkzeuge benutzt [Ste07a]. Verschiedene Unternehmen setzen unterschiedliche Methoden zur Lösung von Problemen des gleichen Typs ein. Außerdem nutzen Experten auf einem Gebiet vorwiegend die von ihnen favorisierten Werkzeuge, z.B. ist die Entscheidung für ein spezielles CAD-System manchmal eher „philosophisch“ denn technisch-wirtschaftlich motiviert. Daraus resultiert, dass Projektmitglieder Methoden und Werkzeuge falsch, nicht richtig an den tatsächlichen Kontext angepasst oder überhaupt nicht anwenden [Ben01]. Falls Methoden angewendet werden, wird häufig eine unüberschaubar große Anzahl angewendet [Ben01].

In [van08] wird angegeben, dass die Integration unterschiedlicher Methoden die größte Herausforderung darstellt. Einige Ansätze (z.B. [Klä93] [Fra05d]) zeigten bereits, dass verschiedene Methoden in eine integrierte rechnerunterstützte Entwicklungsumgebung umgesetzt werden können. Allerdings sind diese Ansätze begrenzt auf die im Vorfeld implementierten Methoden und es muss ein spezieller Formalismus für die Anwendung erlernt werden.

Einer Idee aus [Dae02] folgend zeigt Abbildung 2.13 a) den Nutzen von Formalisierung als Kurve über dem Grad der Formalisierung. Der maximale Nutzen liegt zwischen einem zu hohen und einen zu niedrigen Grad an Formalisierung.

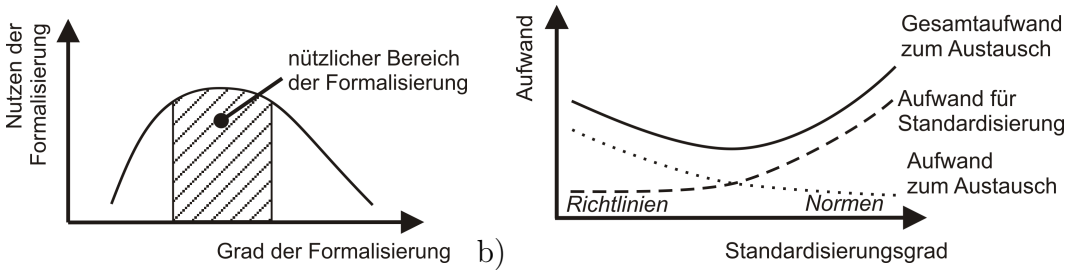


Abbildung 2.13: *Der Nutzen von Formalisierung a) nach [Dae02, S. 249] und b) nach [Avg07, S. 92].*

Ein zu niedriger Grad an Formalisierung liegt z.B. dann vor, wenn Begriffe unterschiedlich gedeutet oder Skizzen auf verschiedene Weise interpretiert werden können. Ein zu hoher Grad an Formalisierung liegt z.B. dann vor, wenn mehr Arbeit auf die Einhaltung der korrekten Form als auf den Inhalt verwendet werden muss (z.B. 3D-Modell und Zeichnungsableitung für einen nur einmal zu verwendenden Blechwinkel). Es ist klar, dass der Verlauf der Kurve stark von der Art des betrachteten Projekts und dem Zeitpunkt im Projektverlauf abhängt. Beispielsweise liegt bei einer geringen Anzahl der involvierten Fachbereiche und Projektmitarbeiter die Kurve weiter links, d.h. eine geringere Formalisierung ist für eine effektive Bearbeitung ausreichend. Sollen hingegen Daten später von bisher nicht Beteiligten weiterverwendet werden können, so müssen diese entsprechend formal aufgebaut sein. Aus diesem Grund wird in Abbildung 2.13 a) ein nützlicher Bereich um das Optimum der Kurve gelegt.

Abbildung 2.13 b) zeigt einen ähnlichen Ansatz. In [Avg07] wird der Aufwand zur Standardisierung und der Aufwand zum Austausch von Informationen getrennt über dem Standardisierungsgrad betrachtet. Der Aufwand zur Standardisierung ist bei Leitlinien relativ gering während der Aufwand zum Austausch der Informationen relativ hoch ist. Bei hohem Standardisierungsgrad (charakterisiert durch Normen) kehrt sich das Verhältnis um. Der minimale Gesamtaufwand findet sich demnach zwischen einem zu niedrigen und einem zu hohen Standardisierungsgrad wieder.

Als zusätzliche Information kann diesem Diagramm folgende Aussage entnommen werden: Wenn der Aufwand zur Standardisierung bereits unternommen wurde (z.B. eine Norm wurde bereits erarbeitet), kann der Nutzen dieser Standardisierung (z.B. Anwenden einer bekannten Norm) fast vollständig abgeschöpft werden. Der Nutzen ist um so höher, je verbreiteter die Standardisierung ist. Eine Notation oder Norm, die bereits in der Ausbildung der Entwickler erlernt wurde (z.B. die Regeln des technischen Zeichnens), bringt bei der Durchführung des aktuellen Projekts demnach fast ausschließlich Nutzen.

Kruse gibt an, dass eine hohe Kommunikationsqualität nur bei einer geringen Varianz der Darstellungsarten erfolgen kann, da die Darstellung dann von allen Projektmitarbeitern schnell und einfach verstanden werden kann und Missdeutungen der Information abnehmen [Kru96, S. 121]. Gleichzeitig fordert er eine mittlere Informationstiefe, da ein Zuviel an Information die Übersichtlichkeit und damit das Verständnis behindert. Allerdings soll die Darstellung auf größere Informationstiefen ausgeweitet werden können (z.B. durch spezifische Sichten auf das System).

Als weitere Einflussfaktoren sind die Akzeptanz durch die Stakeholder, erforderliche Methodenkenntnis, Spezifikationsebene, Art des Systems (z.B. technisch, betriebswirtschaftlich), Komplexität des zu beschreibenden Sachverhalts, Konsistenz bzgl. späteren Änderungen, Eindeutigkeit und Methodenfamilien (z.B. SysML, Unified modelling Language (UML)) zu berücksichtigen [Rup07, S. 218f].

2.2.7 Organisatorische Aspekte

Begrenzte Ressourcen, dynamische Randbedingungen und Zielkonflikte beeinträchtigen die Leistung und Produktivität [Ben01]. Systematische Entscheidungsfindungsprozesse und das Management von Projektzielen sind große Schwachstellen [Zan99], d.h. die klare Definition von Zielen und ggf. deren Anpassung bei sich ändernden Randbedingungen sind Schlüsselfaktoren (vgl. Abschnitt 2.1.3 und Abschnitt 2.1.4).

Auf Grund von Hierarchien (z.B. Lenkungskreis, Projektmanagement und Projektmitarbeiter), unterschiedlichen Fachbereichen (z.B. Maschinenbau, Elektrotechnik, Informationstechnologie) und verschiedener Standorte (z.B. unterschiedliche Unternehmen oder Abteilungen) entstehen eine Reihe unabhängiger „Wissensinseln“. Jede dieser Wissensinseln nutzt eine spezifische Teilmenge des gesamten Projektwissens. Häufig sind sie nicht ausreichend miteinander vernetzt [Gau01]. Es bestehen im Allgemeinen keine hinreichenden organisatorischen Gegebenheiten, die es ermöglichen, im Projekt vorhandenes Wissen an die Stellen zu befördern, wo es gebraucht wird.

Die Organisation eines multidisziplinären Kooperationsprojekts für die Entwicklung komplexer Produkte erfordert einen großen Aufwand an Projektplanung bevor das eigentliche Projekt startet. Wichtige zu klärende Punkte sind die Zeit- und Kostenpläne. Außerdem sind das Entwicklungsrisiko und die Möglichkeit Änderungen zu berücksichtigen (z.B. geänderte Kundenwünsche, Gesetzeslage, verfügbare Technologien) Haupteinflussfaktoren für den Erfolg eines Projektes. Änderungen und Risiko sollen vorhergesagt, beeinflusst, verfolgt und verarbeitet werden. Weitere wichtige organisatorische Rahmenbedingungen sind die Art der Entwicklungsaufgabe (Neuentwicklung oder Altsystemerweiterung), Entscheidungsprozesse, Projektbudget, Anzahl der Stakeholder, Anzahl der Projektmitarbeiter, Vertragsmodell, zeitliche Verfügbarkeit und räumliche Verteilung der Stakeholder [Rup07, S. 75]. Die frühzeitige Klärung der organisatorischen Rahmenbedingungen und deren Planung kann spätere Probleme vermeiden helfen („*Frontloading*“). Allerdings müssen alle Ziele und Pläne ständig auf ihre Erfüllung hin überwacht und auf ihre Zweckmäßigkeit gegenüber den aktuellen Randbedingungen überprüft werden (vgl. auch Abschnitt 2.1.3 und Abschnitt 2.1.4).

2.3 Modularität und Flexibilität von Produkten

Im Folgenden soll auf einen weiteren wichtigen Aspekt der Produktentwicklung eingegangen werden, der bereits in den frühen Phasen der Produktentwicklung—insbesondere bei der Klärung und Verarbeitung von Anforderungen—explizit berücksichtigt werden muss: Die anzustrebende Modularität und Flexibilität der Produkte. Durch die Berücksichtigung mehrerer Varianten und verschiedener Nutzungsweisen des Produktes bereits in der Entwicklung, erhöht sich die An-

zahl und Komplexität der Anforderungen gegenüber Standardentwicklungen. Im Rahmen dieser Arbeit wird sich auf die Entwicklung von Baukastensystemen beschränkt, wie sie z.B. bei der Entwicklung von Parallelrobotern (vgl. Abschnitt 5.2) Anwendung finden.

Die Entwicklung von Baukastensystemen wird häufig dann in Erwägung gezogen, wenn eine größere Anzahl von unterschiedlichen Varianten eines Produktes oder ein gewisser Grad an Flexibilität erreicht werden soll [Lie04] [Fir03]. Baukästen sind demnach nicht nur in der Großserie (z.B. Automobilbau) effizient einzusetzen, sondern auch in der Kleinserie [Jes96] [Fra02b] [Ste09c] (z.B. Parallelroboter). Hier werden Produkte an sehr spezielle Aufgaben angepasst, so dass jedes Produkt beinahe ein Unikat ist. Durch Baukastensysteme werden auch hier Kosten reduziert, indem Skaleneffekte beim Einkauf und in der Fertigung ausgenutzt werden. Außerdem wird die Produktentwicklungszeit je Variante drastisch reduziert und die Qualität der Produkte durch bereits erprobte Komponenten erhöht. Es ist wichtig, die interne Variantenvielfalt klein zu halten und dennoch eine breite externe Variantenvielfalt bereitzustellen [Fir03].

Nachteile von Baukastensystemen sind ein erhöhter Entwicklungsaufwand und damit eine längere Entwicklungsdauer der Erstentwicklung. Außerdem werden die unternehmensinternen Strukturen und Prozesse deutlich komplexer. Es gibt eine große Anzahl unterschiedlicher Arten von Baukastensystemen und verschiedener Methoden für ihre Entwicklung.

Baukastensysteme wurden in den vergangenen Jahren immer wieder in verschiedene Klassen eingeteilt [Nas53, S. 87f] [Bor61, S. 24f] [Kol94, S.338ff] [Koh97, S. 39ff] [Ehr05, S. 338ff] [Ren07, S. 56ff]. Die wichtigsten Unterscheidungsmerkmale wurden aus der Literatur herausgearbeitet und sind in Tabelle 2.1 systematisch zusammengefasst dargestellt (vgl. auch [Mey09]). Beispielsweise wird im Baukastencharakter zwischen Anwender- und Herstellerbaukästen unterschieden.

Tabelle 2.1: *Zusammenstellung der wichtigsten Unterscheidungsmerkmale für Baukastensysteme nach [Nas53], [Bor61], [Kol94], [Koh97], [Ehr05] und [Ren07].*

Baukastencharakter	**Reinheit**	**Verbindungsstruktur**
Herstellerbaukasten	*Reinsystem*	*kettenförmig*
Anwenderbaukasten	*Mischsystem*	*sternförmig*
Ausdehnung	**Konkretisierungsgrad**	*baumförmig*
Lieferantenbaukasten	*abstrakte Bausteine*	*allgemein (Mischstrukturen)*
OEM-Baukasten	*konkrete Bausteine*	**Anpassungsfähigkeit**
Industriebaukasten	**Position des Bausteins**	*nach Abmessungen*
Systemabgrenzung	*strukturgebunden*	*nach wechselnden Aufgaben*
offene Systeme	*modular*	*in der Anordnung*
geschlossene Systeme	**Dimension**	*an Funktionsgrößen*
Verflechtung	*eindirektional*	**Nutzungsdauer**
horizontale Verflechtung	*zweidirektional*	*zeitweilig*
vertikale Verflechtung	*dreidirektional*	*dauernd*

Bei ersterem soll der Anwender das Produkt als Baukasten erkennen und nutzen können. So kann z.B. ein vorhandenes Regalsystem nach dem Umzug in eine neue Wohnung in veränderter Anordnung wieder aufgebaut werden (vgl. Anpassungsfähigkeit: *in der Anordnung*). Bei Herstellerbaukästen ist der Baukastencharakter im Allgemeinen nur für den Hersteller sichtbar. Die externen Varianten sind für den Kunden unterschiedliche Produkte (z.B. Industriegetriebe). Je nach Entwicklungszielen und konkreten Randbedingungen müssen für das zu entwickelnde Baukastensystem die Merkmale festgelegt werden.

In [Fir03] werden über 40 verschiedene Methoden und Methodiken zur Baukastenentwicklung aus der Literatur vorgestellt. Neuere Ansätze finden sich z.B. in [Kle04], [Pul04], [Sek05], [Ren07], [Ble08a] und [Gun08]. Viele dieser Ansätze unterscheiden sich nur in ihrer konkreten Ausgestaltung. Als allgemeiner Ansatz für die Baukastenentwicklung lassen sich nach *Pimmler* drei grundlegende Schritte identifizieren [Pim94] (vgl. auch Descartes [Des73, S. 108ff]):

1. Zerlegen des Systems in einzelne Elemente.
2. Erfassen der Beziehungen zwischen den Elementen.
3. Vereinigen der Elemente zu strukturellen Blöcken und Arbeitspaketen.

In der Konzeptphase kann das Produkt in Form von Funktionsstrukturen (z.B. [Fra76] [Pah07]) beschrieben werden. Eine Funktionsstruktur besteht dabei immer aus Funktionselementen und Flüssen zwischen ihnen. Bereits hier können die Funktionen zu übergeordneten Blöcken vereinigt werden. Wenn nun die Funktionen auf Funktionsträger konkretisiert werden, bilden diese Bausteine oder Module. Die Art der Bausteine wird dabei nach Art der Funktionen klassifiziert (z.B. Grund- oder Hilfsfunktion) und weiter in Muss- oder Kann-Bausteine unterschieden (z.B. in [Pah07, S. 664]). Außerdem müssen die Schnittstellen standardisiert werden.

Auf Grund der großen Anzahl von Umsetzungsmöglichkeiten für Baukastensysteme ist es entscheidend sich vor der Baukastenentwicklung über die zu erreichenden Ziele im Klaren zu sein. *Renner* gibt in seiner Dissertation [Ren07] sieben Stoßrichtungen bzw. Ziele der Entwicklung an (vgl. Abb. 2.14). Diese sind allerdings weder vollständig noch überlappungsfrei und eignen sich daher nicht für eine Klassifizierung. Dennoch bieten sie einen Startpunkt für eine Zieldefinition und die Festlegung von Strategien für deren Umsetzung.

In Abb. 2.14 sind zu jeder Stoßrichtung verschiedene Beispiele angegeben, die eine Umsetzung dieser Ziele ermöglichen. Im dargestellten Netzdiagramm ist für die zwei Beispiele Sitz und Klimakompressor eines Pkw eine Gewichtung dargestellt. Es wird beschrieben, dass für einen Klimakompressor eine Erhöhung der Flexibilität eher unwichtig eingeschätzt wird, während z.B. die Reduktion der Variantenanzahl einen Schwerpunkt bildet. Ein Sitzbaukasten soll hingegen eine hohe Flexibilität bieten und muss die Variantenanzahl nicht reduzieren. Unterschiede an einem Sitz werden im Gegensatz zu einem Klimagerät vom Kunden

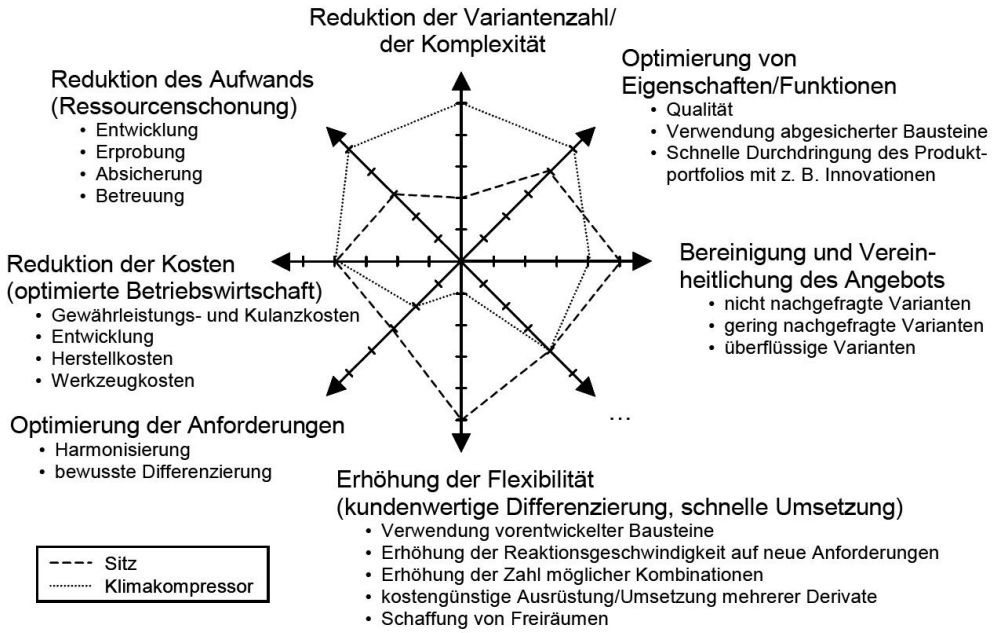


Abbildung 2.14: *Stoßrichtungen der Baukastenentwicklung [Ren07, S. 118].*

direkt als Diversifikationsmerkmale wahrgenommen. Es wird davon ausgegangen, dass der gleiche Sitz in unterschiedlichen Fahrzeugen und Fahrzeugausstattungen von Kundenseite nicht akzeptiert wird. Ein Klimakompressor kann hingegen in jedem Fahrzeug identisch sein, solange er ausreichend kühle Luft liefert.

Firchau [Fir03, S. 14] und *Lindemann* [Lin01, S. 76] weisen darauf hin, dass nur eine flexibel auf die Erfordernisse der konkreten Situation angepasste Nutzung unterschiedlicher Methoden zu einem erfolgreichen Baukastenkonzept führt. Die weiteren Schritte in *Renners* Vorgehen folgen diesem Hinweis und bieten einen allgemeinen Rahmen für die Anwendung verschiedener Methoden. Der oben beschriebenen Themenpriorisierung folgt eine Analysephase in der Anforderungen, Funktionen und Vernetzungen betrachtet werden (vgl. Schritt 1 und 2 [Pim94]). In der anschließenden Synthesephase werden die Baukastenkernelemente definiert und mögliche Szenarien entworfen (vgl. Schritt 3 [Pim94]). Als dritten Schritt sieht *Renner* eine geeignete technische und wirtschaftliche Bewertung (z.B. in Form eines Businessplans) vor. Die eigentliche Umsetzung (d.h. Gestaltung von Bauteilen und Schnittstellen) ist dieser Konzepterstellung nachgelagert. Innerhalb der drei Schritte der Konzepterstellung sind häufige Vor- und Rücksprünge vorgesehen.

2.3.1 Anforderungen in der Baukastenentwicklung

Wie in Abschnitt 2.1 beschrieben, ist das Anforderungsmanagement für komplexe interdisziplinär zu entwickelnde Produkte sehr aufwändig und die Anforderungs-

dokumentation sehr umfangreich. Kommt der Wunsch nach einer Baukastenentwicklung hinzu, so müssen nicht nur Anforderungen an ein Produkt, sondern an eine Anzahl von verschiedenen aus dem Baukasten zu generierenden Varianten berücksichtigt werden. Auf das Anforderungsmanagement ist somit bei der Entwicklung von Baukastensystemen ein besonderes Augenmerk zu richten [Ren07, S. 128f]. Ein Beispiel für solche Produkte sind die in Abschnitt 5.2 als komplexes Anwendungsbeispiel genauer beschriebenen Parallelroboter. Im Folgenden wird zur Verdeutlichung bereits auf Beispiele aus diesem Anwendungsbereich zurückgegriffen. Es lassen sich Anforderungen an das Baukastensystem (z.B. *Varianten sollen schnell zu konfigurieren sein*), an die Baukastenprodukte (z.B. *Das Robotersystem soll eine Beschleunigung von* $a_{TCP} = 10 \cdot g$ *am Tool Center Point (TCP) erreichen*) und vom Baukastensystem auf das Produkt zurückwirkende Anforderungen (z.B. *Die Gelenke sollen standardisierte Schnittstellen bereitstellen*) unterscheiden (vgl. [Ren07, S. 102/130]).

Eine spezielle Tätigkeit im Rahmen der Anforderungsverarbeitung (vgl. Abschnitt 2.1.9) ist das von *Renner* als Anforderungsharmonisierung [Ren07, S. 130f] bezeichnete Verarbeiten von gleichen Anforderungen, die für unterschiedliche Varianten verschiedene Zielbereiche angeben. Beispielsweise soll der vom Gestell umschlossene Roboterarbeitsraum für eine Pick-and-Place-Aufgabe in eine Richtung möglichst lang ($> 1,5\,\mathrm{m}$) und für eine Montageaufgabe in alle Richtungen etwa gleich allerdings kürzer ($> 0,5\,\mathrm{m}$) sein. Dieser Widerspruch muss „harmonisiert“ werden (vgl. [Bec06] [Sur09]). Es kann ein Kompromiss gefunden werden (z.B. $1,0\,\mathrm{m}$ als Punktforderung), der allerdings für die meisten Varianten eine Unter- oder Übererfüllung darstellt. Je nach Aufgabenstellung ist auch eine Abstufung denkbar (z.B. $0,5\,\mathrm{m}$, $1,0\,\mathrm{m}$ und $1,5\,\mathrm{m}$). Dabei ist zu beachten, dass die verschiedenen Abstufungen unterschiedlicher Varianten nicht in allen Kombinationen sinnvoll sind. Hier ist das frühzeitige Festlegen von Bedingungen (z.B. unverträgliche Anwendungsfälle) sinnvoll, um eine sogenannte kombinatorische Explosion [Fra98] zu vermeiden. Außerdem ist zu untersuchen, welchen Einfluss eine Anforderung auf das Gesamtsystem ausübt. Damit lässt sich die Frage beantworten: Zieht der Übergang von der niedrigsten zur höchsten Abstufung weitere Maßnahmen nach sich? Im angegebenen Beispiel wird ein Träger bei $1,5\,\mathrm{m}$ Stützweite deutlich größere Verformungen erfahren als ein Träger bei $0,5\,\mathrm{m}$ Stützweite. Es ist zu überprüfen, ob die hierdurch beeinflusste Robotergenauigkeit in beiden Fällen noch ausreicht oder etwa andere Trägergeometrien, zusätzliche Stützen oder adaptronische Komponenten in das Baukastensystem integriert werden müssen.

Zur systematischen Bearbeitung der Anforderungsharmonisierung wird eine Halbmatrix vorgeschlagen, die sämtliche Anforderungen in Beziehung zueinander setzt [Ren07, S. 130]. Eine solche Matrix wird unter den Bezeichnungen Anforderungs-, Konsistenz- oder Zielkonfliktmatrix sowie als Dach des House of Qualities (HoQ) im Quality Function Deployment (QFD) auch von verschiedenen weiteren Autoren verwendet (z.B. [Kru00, S. 100], [Lin05, S. 101], [Ste06b]).

Die Beziehungen sollen nach Zielkonflikten (die Verbesserung der Erfüllung einer Anforderung bewirkt eine Verschlechterung einer weiteren Anforderung) und Divergenzen bzw. Anforderungsspreizungen (zwei Anforderungen beziehen sich auf denselben Sachverhalt besitzen aber andere Wertebereiche) untersucht werden. In der Matrix bilden sich ggf. Cluster von Beziehungen, die bei der weiteren Entwicklung zusammenhängend betrachtet werden sollen.

Für die Softwareentwicklung schlägt *Rupp* vor, eine gemeinsame Anforderungsbeschreibung für identische Teile und einige spezialisierte Spezifikationsteile für Abweichungen zu erstellen [Rup07, S. 435]. *Pohl* bezeichnet einen ähnlichen Ansatz als *feature trees* mit *variation points* [Poh05, S. 99ff], wobei die *variation points* als Anforderungsspreizungen aufgefasst werden können. Weitere Ansätze finden sich in der Software-Produktlinienentwicklung und für UML-Use-Cases.

Im Sinne eines Anforderunsmanagements zur Baukastenentwicklung unterscheidet *Rupp* weiterhin in Entwicklung *durch* Wiederverwendung (Application-Requirements-Engineering) und Entwicklung *zur* Wiederverwendung (Domain-Requirements-Engineering) [Rup07, S. 437ff]. Die Entwicklung zur Wiederverwendung ist analog zur Entwicklung von Baukastensystemen und umfasst die folgenden Aufgaben:

- Detaillierung der Merkmale aus dem Produktmanagement in Produktlinienanforderungen.
- Definition von Gemeinsamkeiten und Variabilität.
- Explizite Repräsentation der Variabilität.

Die Entwicklung durch Wiederverwendung zeigt hingegen Analogien zur Konfiguration einer Produktvariante aus einem Baukasten und umfasst die Aufgaben:

- Kommunikation der Produktlinienanforderungen und -variabilität.
- Spezifikation von Anforderungen, die nicht vollständig durch Wiederverwendung definiert werden können.
- Unterstützung von Trade-Off-Entscheidungen.
- Rückkopplung für das Domain-Engineering.

Zur Wiederverwendung eignen sich im Rahmen des Anforderungsmanagements demnach [Rup07, S. 456]: Abschnitte des Benutzerhandbuchs, nicht-funktionale und funktionale Anforderungen, Fragen und Interviewdokumente, Testszenarien, Analysemodelle, Abnahmekriterien, Begriffsdefinitionen, Prozess- und Rollenbeschreibungen. Die Wiederverwendbarkeit ist umso besser [Rup07, S. 457]:

- je höher der Spezifikationslevel ist,
- je „konstanter“ das Fachgebiet ist,

- je unabhängiger die Einbindung in einen Ablauf ist,
- je abstrakter und technologieunabhängiger die Beschreibung ist und
- je konstanter die Ausprägungen des Inhalts sind.

Elemente, die diese Kriterien erfüllen sind jedoch durch einen vergleichsweise geringen Informationsgehalt gekennzeichnet. Sie müssen im speziellen Projekt stets weiter konkretisiert und spezifisch angepasst werden.

2.3.2 Änderungen in Produktentwicklung und Betrieb

Wie bereits in Abschnitt 2.1 erwähnt, gehören Iteration zu jedem Produktentwicklungsprozess (z.B. weil konkrete Informationen erst in späteren Phasen vorliegen) und Änderungen im Betrieb gehören zur Anwendung beinahe jeden Produkts (z.B. Anpassung des Autositzes bei Fahrerwechseln). Der Ansatz des XP [Bec00] geht davon aus, dass die Implementierung von Anforderungen in eine Software einer flachen Aufwandskurve folgt, d.h. der Aufwand zur Implementierung ist kaum abhängig vom Zeitpunkt der Entdeckung bzw. Änderung der Anforderung. Diese Aussage erscheint allerdings nur für funktionale Anforderungen im objektorientierten Programmieren haltbar. In Entwicklungsprozessen für mechatronische Systeme kann hingegen die bekannte „Zehnerregel“ als Richtwert angesehen werden. Hier wird davon ausgegangen, dass der Aufwand (die Kosten) für eine Änderung beim Übergang in die nächste Produktlebensphase jeweils um den Faktor 10 ansteigt. Man kann sich vorstellen wie aufwändig sich bspw. eine Geometrieänderung ausnimmt, wenn bereits Werkzeuge gebaut und Maschinen beschafft sind.

Änderungssituationen während der Produktentwicklung lassen sich in drei Klassen einteilen (basierend auf einer Klassifikation in [Rup07, S. 494f]):

- unvermeidbare, immer wieder vorkommende Änderungen (Iterationen),
- nachträgliche, auf bestimmten Entscheidungen des Auftraggebers beruhende Änderungsverlangen,
- vom Auftraggeber „vergessene“ und vom Auftragnehmer nicht als solche erkennbaren Nachträge (Korrekturen).

Die Iterationen der ersten Klasse sind kaum vermeidbar, da mit fortschreitender Konkretisierung erst bestimmte Randbedingungen geklärt werden können. Dadurch müssen bestimmte Aktivitäten zunächst mit Schätzwerten durchgeführt werden, um später mit exakten Werten belegt werden zu können. Beispielsweise sind zur Auswahl eines bestimmten Bauteils die am Bauteil auftretenden Kräfte notwendig. Die Kräfte hängen allerdings entscheidend von der Masse eben diesen Bauteils ab. So muss zuerst eine Größenordnung bestimmt, ein Bauteil ausgewählt und schließlich überprüft werden. Durch eine geeignete Prozessgestaltung

können diese Änderungen vergleichsweise überschaubar gehalten werden.

Die zweite Klasse beruht auf Änderungen in den Randbedingungen. Diese führen dazu, dass zuvor festgelegte Anforderungen neue Werte annehmen. Beispielsweise kann aus marktpolitischen Gründen der Start Of Production (SOP) vorverlegt werden, so dass die ursprüngliche Terminplanung nicht mehr gilt. In diese Klasse können darüber hinaus auch Änderungen eingeordnet werden, die als Variantenentwicklung auf einem zuvor entwickelten Produkt basieren und geänderte Randbedingungen (z.B. anderer Kunde, Facelift) berücksichtigen.

Die dritte Klasse berücksichtigt Aspekte, die z.B. auf Grund einer unvollständigen Aufgabenklärung nicht in die Entwicklung eingeflossen sind. Hierbei handelt es sich meist um implizite Anforderungen, die dem Kunden zwar bekannt sind, danach gefragt aber nicht einfallen.

Änderungen die während der Betriebsphase eines Produktes auftreten, beruhen im Allgemeinen auf veränderten Randbedingungen der Aufgabe oder der Umgebung. Beispielsweise können sich die zu handhabenden Objekte in einer Produktionsanlage ändern, wenn ein neues Produkt eingeführt wird. In einem solchen Fall muss entweder die Produktionsanlage durch eine neu entwickelte ausgetauscht werden oder sie wird an die geänderten Randbedingungen angepasst, ist also bis zu einem gewissen Grad flexibel. Es lassen sich zwei Arten der Anpassung unterscheiden:

- statische Anpassung: Das Produkt wird aus dem eigentlichen Betrieb entfernt, einige Teile werden ausgetauscht oder die Anordnung wird verändert (z.B. CPU tauschen, um einen leistungsfähigeren Rechner zu erhalten).

- dynamische Anpassung: Das Produkt wird während des Betriebes an die neuen Randbedingungen angepasst (z.B. Höheneinstellung eines Schreibtischstuhls zur Anpassung an einen größeren/kleineren Nutzer).

Bischoff fasst in [Bis07] einige Gestaltungsichtlinien zur Entwicklung flexibler Produkte zusammen. Es werden 24 Regeln in einem Katalog angegeben, der neben einer Beschreibung Abbildungen zur nicht-flexiblen und flexiblen Gestaltung an einem Beispiel gegenüberstellt. Die Wirkweise der einzelnen Regeln wird in drei Kategorien unterschieden [Bis07, S. 6]:

- Auswirkung einer Änderung wird reduziert.

- Aufwand, der bei/durch eine Änderung entsteht wird reduziert.

- Auftrittswahrscheinlichkeit von Änderungen am Produkt wird reduziert.

Beispielsweise wird die Regel „Differentialbauweise statt Integralbauweise anwenden“ den ersten beiden Kategorien zugeordnet. Regeln, die eine Überdimensionierung fordern, werden der dritten Kategorie zugeordnet.

2.4 Bewerten und Testen

Das Bewerten und Testen nimmt einen wichtigen Stellenwert in der Produktentwicklung und Projektüberwachung ein (vgl. Abschnitt 2.1.4). Im Wesentlichen soll das Entwicklungsrisiko minimiert werden, indem einerseits Fehlerquellen beseitigt werden und andererseits auf die Markterfordernisse hin entwickelt wird. Letzteres stellt sicher, dass nicht am Markt vorbei entwickelt wird, d.h. Produkte entstehen, die die Ziele und Anforderungen der Kunden erfüllen und daher auch gekauft werden. Das Testen—wie es in dieser Arbeit verstanden werden soll—stellt somit das Spiegelbild zu den Anforderungen dar, weshalb ihm hier ein eigener Abschnitt eingeräumt wird.

Im Bereich der Projektüberwachung müssen Zeit, Kosten und weitere Ressourcen überwacht werden. Hier gibt es umfangreiche Literatur, Methoden und Softwareunterstützung unter dem Stichwort Projektmanagement. In der Produktentwicklung gibt es ebenfalls zahlreiche Methoden, die in unterschiedlichen Phasen der Produktentwicklung angewendet werden. Dabei geht es darum, die Zielerreichung, Produkteigenschaften und Produktmerkmale zu bewerten. Insbesondere bei der Entwicklung von Produkten mit einem hohen Anteil an Software und Elektronik oder bei einem hohen Risiko für Leib und Leben sind zum Teil aufwändige und stark formalisierte Testverfahren notwendig. Beispiele hierfür sind die Failure Mode and Effects Analysis (FMEA) und Fault Tree Analysis (FTA) für die Identifizierung möglicher Fehlerquellen und die Bewertung des Risikos. Dabei treten Risiken zum einen durch eine unerwartete Nutzung des Systems ein. Zum anderen sind Risiken auf unzureichende Qualität des Produktes zurückzuführen. Auf Basis der Risiken bzw. deren Priorisierungen wird festgelegt, wann welche Anforderungen überprüft werden müssen und wie hoch der jeweils gerechtfertigte Arbeitsaufwand ist. Anschließend werden geeignete Testverfahren für bestimmte Komponenten und Aktivitäten festgelegt. Beispiele für Testverfahren sind Designreview, Stellungnahme, Inspektion, Prototyp, Simulationsmodell, Analysemodell, Walkthrough und Agile Specification Quality Control [Rup07, S. 311ff].

Grundsätzlich benötigt jedes Bewertungsverfahren Bewertungskriterien. Diese werden sinnvollerweise aus den Anforderungen gewonnen und können entsprechend in Anforderungsmodellen und -listen verknüpft werden (z.B. [Kic95, S. 49] [Sch01, S. 109f]). Dabei eignen sich in frühen Entwicklungsphasen Anforderungen auf hohem Abstraktionsniveau, während in späteren Phasen detaillierte Anforderungen herangezogen werden. Es werden Fest- und Mindestforderungen genutzt, um untaugliche Lösungen auszuscheiden. Mindestforderungen und Wünsche werden genutzt, um eine Reihung unterschiedlicher Lösungen zu erreichen. Zum Ende der Entwicklung müssen alle Anforderungen überprüft sein. Insbesondere die bei Auftragsentwicklungen vertraglich festgelegten Anforderungen müssen als Abnahmekriterien getestet werden. *Rupp* gibt folgende Qualitätsmerkmale für Abnahmekriterien an [Rup07, S. 326f]:

- Testbarkeit:
  - Durchführbarkeit,
  - Messbarkeit und
  - Reproduzierbarkeit,
- Vollständig bezüglich der Anforderung und
- Minimal bezüglich der Anforderung.

Die Abnahmekriterien werden in Testszenarien oder Testfällen gebündelt [Rup07, S. 344ff]. Ziel ist es, mit möglichst geringem Aufwand einen möglichst großen Teil der Abnahmekriterien im selben Testfall zu überprüfen. Ein Testfall besteht allgemein aus den drei Komponenten Vorbedingungen, Durchführung und Nachbedingungen.

Kapitel 3

Modellierung im Produktentwicklungsprozess

Im vorangegangenen Kapitel wurden die wesentlichen Aspekte einer interdisziplinären Produktentwicklung näher betrachtet. Dazu wurden verschiedene allgemeine und spezielle Produktentwicklungsprozesse miteinander verglichen. Anschließend wurden typische Schwachstellen der verteilten Produktentwicklung herausgearbeitet. Besonderes Augenmerk lag auf der Entwicklung von modularen und flexiblen Produkten. Schließlich wurde betrachtet, dass Konzepte und Produkte während des Entwicklungsprozesses bewertet und getestet werden müssen.

Im Folgenden soll untersucht werden, wie eine entwicklungsbegleitende Modellierung verwirklicht werden muss, um die in Abschnitt 2 identifizierten Schwachstellen zu entschärfen. Dabei liegt das Hauptaugenmerk auf der Modellierung der Anforderungen als Kern einer erfolgreichen Produktentwicklung.

Im Produktentwicklungsprozess werden die verschiedensten Modelle verwendet. Die Definitionen, was ein Modell ist und was es ausmacht sind vielfältig und hängen stark vom Ziel der Modellierung ab. Ein früher Versuch einer allgemeingültigen Definition stammt von *Franke* [Fra76, S. 28]:

> *Gegeben ist ein reales System S (z.B. ein System gegenständlicher Objekte, ein System realer Ereignisse o.ä.) und eine Menge M beliebiger Objekte (z.B. Zeichen oder Symbole, grafische Elemente, Worte, Gegenstände). Eine umkehrbar eindeutige Abbildung eines Teilsystems $S_t \subset S$ in die Menge M heißt dann Modell von S.*

Zwanzig Jahre später beschreibt *Ort* in seiner Dissertation Modelle so [Ort98, S. 11]:

> *Modelle geben die Zusammenhänge zwischen realen Gegebenheiten in einer abstrakten Form wider. Als ein Abbild der Wirklichkeit besitzen Modelle Vereinfachungen, die auf der einen Seite zu einem Verlust der Wirklichkeitstreue führen, auf der anderen Seite aber eine Transparenz und Beherrschbarkeit der realen Zusammenhänge mit sich bringen.*

Avgoustinov definiert in seinem Buch Modelle folgendermaßen [Avg07, S. 9]:

> *Model is a purpose-dependent, finite, simplified, but still adequate representation of whatever is modelled, allowing us to abstract from its unimportant properties and details and to concentrate only on the most specific and most important traits.*

Die drei genannten Definitionen enthalten die Aspekte Abstraktion, Formalisierung und Vereinfachung, um Transparenz und eine Konzentration auf die wesentlichen Eigenschaften zu erreichen. Dabei muss laut *Franke* die Abbildung eindeutig und umkehrbar sein. *Ort* weist weiter darauf hin, dass ein Modell die Kontrollierbarkeit der komplexen Realität erlaubt. Als Motto ließe sich ableiten: „Wir können nicht kontrollieren, was wir nicht verstehen!“

Ein Modell soll den zu modellierenden Gegenstand repräsentieren, unwichtige Details ignorieren (abstrahieren) und eine pragmatische Anwendung erlauben. Der Zweck der Modellbildung ist es, den Modellierungsgegenstand (Prozess, Tätigkeit, Gegebenheit, Bauteil) besser zu verstehen und eine Basis für Diskussionen und Informationsaustausch zu schaffen. Durch Modelle ist es möglich unterschiedliche Lösungen einer Entwicklungsaufgabe zu analysieren, ihr Verhalten und ihre Eigenschaften vorherzusagen und miteinander zu vergleichen. Dazu muss das Modell eine festgelegte Architektur und Gliederung aufweisen sowie Beziehungen zwischen den enthaltenen und zu externen Komponenten abbilden (vgl. [Avg07, S. 10]).

In Abb. 3.1 sind die drei wesentlichen Aspekte der Modellbildung dargestellt: Ein Modellierer nimmt den Modellierungsgegenstand als Eingabe und liefert als Ausgabe ein Modell. Die Tätigkeit der Modellbildung berücksichtigt dabei den aktuellen Blickwinkel des Modellierers auf den Modellierungsgegenstand, sowie seine Vorbildung und Denkmuster. Das bedeutet, dass Modelle nur dann eindeutig verstanden werden können, wenn der Betrachter die gleiche Vorbildung und zumindest ähnliche Denkmuster verfolgt wie der Modellierer. Ist das nicht

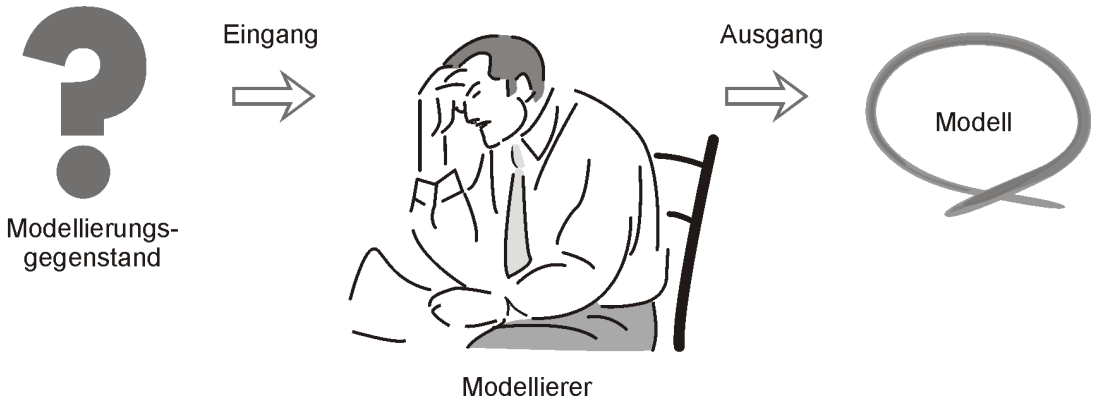


Abbildung 3.1: *Die drei Aspekte der Modellbildung, nach [Avg07, S. 9].*

der Fall kann das Modell entweder gar nicht oder falsch interpretiert werden. Beispielsweise ist die Interpretierbarkeit der Arecibo-Botschaft (Radiowellensignal), der Pioneer-Plaketten (Platten an Bord der Raumsonden Pioneer 10 und 11) und der Voyager Golden Record (Schallplatten an Bord der Raumsonden Voyager 1 und 2) umstritten. Diese Botschaften enthalten in binärer Codierung bzw. als Strichgrafik Informationen und sollen bei Kontakt mit außerirdischem Leben über die Menschheit informieren. Um die dort abgebildeten Striche z.B. als Repräsentation von menschlichen Körpern erkennen zu können, müssen ähnliche Repräsentationsformen und ähnliche Lebewesen bereits bekannt sein. Ist das nicht der Fall, so wird ein Strich der den Haaransatz oder das Schlüsselbein repräsentieren soll nicht als solches verstanden werden. Außerirdische—sollten sie existieren—würden die Botschaften vermutlich falsch interpretieren.

Weiterhin bildet jedes Modell einen bestimmten Blickwinkel ab. Häufig wird deshalb auch von Partialmodellen gesprochen, da sie immer nur einen Teil der Realität abbilden und nur in ihrer Gesamtheit eine hinreichende Modellierung des Modellierungsgegenstandes erlauben.

Lippardt fasst die aus der Literatur bekannten Schwierigkeiten, die sich bei bildlichen Darstellungen im Konstruktionsprozess ergeben folgendermaßen zusammen [Lip00, S. 7-9]:

- Ungenügende Vereinheitlichung von Darstellungsformen,
- Unzureichende Berücksichtigung von Skizzen durch die Konstruktionswissenschaft,
- Mangel an Produktmodellen hoher Reichhaltigkeit,
- Ungenügende Kenntnis der Eigenschaften und der Leistungsfähigkeit von Modellen und
- Unzureichende Möglichkeit zur Visualisierung rechnerinterner Produktdaten.

Diese Schwierigkeiten decken sich zu großen Teilen mit den Aussagen aus Abschnitt 2.2.5 und Abschnitt 2.2.6 und sollen durch eine geeignete Darstellung der Information in einem Modell entschärft werden.

3.1 Anforderungen an Modelle

Wie bereits dargestellt kann es kein allumfassendes Produktmodell geben, dass alle zu betrachtenden Aspekte darstellt. Vielmehr müssen verschiedene Partialmodelle erstellt werden, die den jeweils interessierenden Sachverhalt betrachten (s. Abschnitt 3.2). Diese Partialmodelle sollten in einen übergeordneten Kontext einbezogen werden, der es erlaubt die Modelle entwicklungsbegleitend zu

erweitern und Daten beim Wechsel der Partialmodellen zu übernehmen bzw. die Beziehungen zwischen Elementen verschiedener Partialmodelle abzubilden. In späteren Entwicklungphasen helfen sogenannte Entwicklungsumgebungen einen konsistenten Austausch von Modelldaten zu gewährleisten (vgl. [Fra05d] [Fra05c] [Ste06b] [Ste07a]). In frühen abstrakten Phasen ist es hingegen sinnvoll eine einheitliche, übergeordnete Modellierungsweise zu nutzen, die möglichst lange und über Fachbereichsgrenzen hinweg eingesetzt werden kann. Aber nach welchen Gesichtspunkten müssen Modelle aufgebaut werden? Welche Merkmale können zu ihrer Beurteilung herangezogen werden?

Tabelle B.1 (im Anhang) gibt eine aus verschiedenen Literaturquellen und den Überlegungen der vorangestellten Kapitel erarbeitete Liste der allgemeinen Ziele wider, die Modelle in einer interdisziplinären Produktentwicklung erfüllen sollen. Im konkreten Fall kommen weitere Ziele bzw. Anforderungen hinzu. Beispielsweise kann der Datenaustausch zu einem speziellen Entwicklungswerkzeug der späteren Entwicklungsphasen (z.B. Catia, Unigraphics) gefordert werden. Diese allgemeinen Ziele bilden die Grundlage für die zu entwickelnde Modellierung.

In Tabelle B.2 (im Anhang) sind wesentliche Merkmale zur Beschreibung und Beurteilung von Modellen zusammengestellt. Hierzu wurden die Beurteilungsmerkmale von *Avgoustinov* ([Avg07, S. 40f]), die sich im wesentlichen auf Softwaremodelle beziehen, zugrunde gelegt, kritisch hinterfragt und durch weitere Merkmale ergänzt. Anschließend wurden die gesammelten Daten für eine allgemeine Beschreibung aufbereitet.

Die Modellmerkmale beeinflussen die Erfüllung der Ziele. Diese Zusammenhänge wurden untersucht und sind in Tabelle B.3 dargestellt. Sie bilden die Grundlage für eine Merkmalsfestlegung hinsichtlich der Modellierungsziele. Insbesondere hat der gewählte Abstraktionsgrad der Modellierung einen entscheidenden Einfluss auf die Zielerfüllung. Viele der Ziele (z.B. vertretbarer Arbeitsaufwand, Durchgängigkeit, Transparenz) lassen sich nur durch die Wahl eines hohen Abstraktionsniveaus voll erfüllen. Andererseits müssen bei einem sehr abstrakten Modell Einbußen hingenommen werden. Beispielsweise sind die Unsicherheiten umso größer, je weniger detailliert das Modell ausgebildet ist. Eine Wellenberechnung nach DIN-743 [DIN00] bietet für die meisten Fälle eine hinreichend genaue Abbildung der Wirklichkeit. Die durch die Modellbildung hervorgerufenen Ungenauigkeiten werden durch die Anwendung von Sicherheitsfaktoren kompensiert. Falls eine Überdimensionierung nicht erlaubt ist (z.B. aus Gründen des Leichtbaus), muss eine genauere Berechnung erfolgen, damit auf große Sicherheitsfaktoren verzichtet werden kann. Bei genau bekannten Randbedingungen kann z.B. eine Finite Elemente Methode (FEM)-Berechnung durchgeführt werden.

Einen weiteren großen Einfluss auf die Zielerreichung hat das Merkmal „Blickwinkel“. Es erscheint sinnvoll ein Vorgehen zu wählen, dass es erlaubt während der Entwicklung verschiedene Blickwinkel auf unterschiedlichen Abstraktionsebenen zu nutzen (z.B. funktionale, Bauraum-, Kostenbetrachtung). Gleichzeitig soll es eine Durchgängigkeit im gesamten Produktentwicklungsprozess gewährleisten.

Dazu werden innerhalb eines übergeordneten Konzepts Partialmodelle für bestimmte Zwecke zu etabliert.

3.2 Verschiedene Partialmodelle der Produktentwicklung

Für eine hinreichend genaue Abbildung des Modellierungsgegenstandes ist in jedem Produktentwicklungsprozess eine Anzahl von verschiedenen Modellen notwendig. Diese—auch Partialmodelle oder P_n-Modelle genannten—Modelle sind durch logisch zusammenhängende Inhalte gekennzeichnet, die mindestens diejenigen Systemeigenschaften der jeweiligen Konkretisierungsstufe darstellen. In ihrer Summe bilden sie das integrierte Produktmodell. [Fra76, S. 36] [Bir80, S. 20] [Gra93, S. 17]

Eine Möglichkeit ist es, die verschiedenen Modelle in die Dimensionen Abstraktionsgrad, Anwendungszeitraum im Produktentwicklungsprozess und Fachbereich zu strukturieren. Eine vollständige Abbildung aller möglichen Modelle in alle drei Dimensionen ist auf Grund der hohen Anzahl und der großen Überlappungsbereiche nicht möglich. In Abb. 3.2 ist daher eine Auswahl allgemein beschriebener Modelle abgebildet. Die Fachbereiche Maschinenbau, Elektronik und Softwareentwicklung werden auf drei Abbildungen aufgeteilt. Jede Abbildung zeigt den qualitativen Zeitverlauf des Produktentwicklungsprozesses auf der Abszisse und den Abstraktionsgrad des Modells auf der Ordinate.

Alle drei Abbildungen zeigen ein Anforderungsmodell auf hohem Abstraktionsniveau, dass über den gesamten Produktentwicklungsprozess „mitläuft“. In den darunterliegenden Schichten befinden sich weitere Modelle, die zum Ende

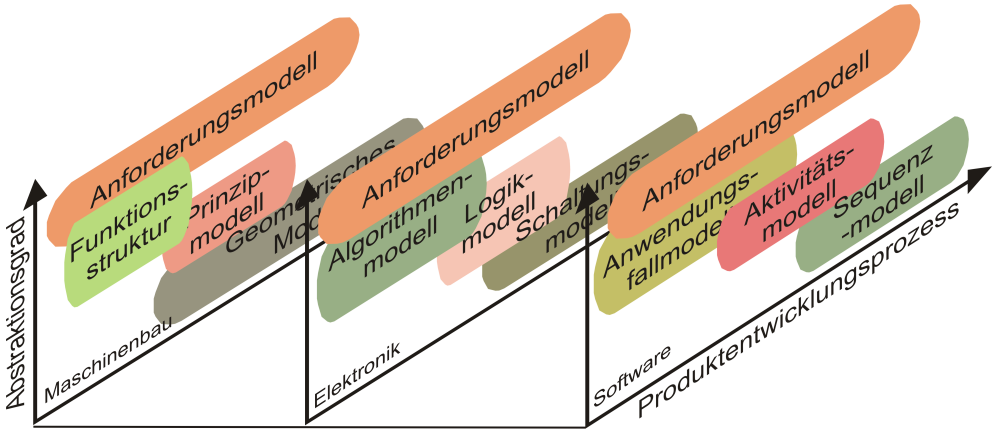


Abbildung 3.2: *Beispiele zur Verwendung von Modellen verschiedener Abstraktionsgrade über dem Produktentwicklungsprozess in den Fachbereichen Maschinenbau, Elektronik und Software.*

des Entwicklungsprozesses hin eine immer geringere Abstraktion vom Endprodukt aufweisen. Dabei werden Modelle über gewisse Phasen überlappend genutzt, z.B. muss die Plausibilität des entworfenen Prinzips durch die Funktionsstruktur überprüft werden. Dazu muss die Funktionsstruktur ggf. angepasst werden, z.B. um notwendige Hilfsfunktionen zu integrieren. Einige Modelle werden auch auf gleichem Abstraktionsgrad parallel genutzt (hier nicht dargestellt), um unterschiedliche Sichtweisen zu untersuchen. Ein Beispiel dafür sind FEM und Computational Fluid Dynamics (CFD), die auf der einen Seite die Struktur (z.B. auftretende Spannungen in einer Tragfläche) und auf der anderen Seite die Strömungsbedingungen (z.B. erzeugter Auftrieb durch die Tragfläche) betrachten. Die ständig wachsende Mächtigkeit aktueller Modellierungswerkzeuge lässt zudem die Modellgrenzen weiter verschwimmen. Beispielsweise verfügen viele CAD-Systeme neben der grafischen Modellierung ebenfalls über integrierte z.B. Mathematik-, FEM-, Mehrkörpersimulation (MKS)-Module [Fra05d] [Fra05c].

Für eine Beschreibung des Produktes auf Systemebene sind im Wesentlichen die im Folgenden beschriebenen Partialmodelle zu berücksichtigen. Sie werden zur Bearbeitung der in Abschnitt 2.1 beschriebenen Bausteine der Produktentwicklung verwendet. Wird die Systemebene verlassen und innerhalb der Bausteine die Konkretisierung durch andere Modellierungssprachen erhöht, so können mehrere der hier abgebildeten Partialmodelle teilweise vereint werden. Beispielsweise kann eine Computer Aided Design (CAD)-Baugruppe die Struktur und (bis zu einem gewissen Grad) die Funktion abbilden.

- Stakeholdermodell (vgl. Abschnitt 2.1.2)
  Im Stakeholdermodell werden sämtliche über den Produktlebenslauf involvierten Rollen erfasst und strukturiert abbildet. Eine mögliche Einteilung erfolgt nach Hersteller, Kunde, Gesetzgeber usw., wobei diese Klassen noch weiter aufgegliedert werden können (z.B. Softwareentwickler, Kinematik Experte). Es entsteht ein hierarchisches System an dessen unterem Ende natürliche Personen stehen. Darüber hinaus sollen Beziehungen zwischen Stakeholdern darstellbar sein, z.B. Kommunikation eines Kunden mit einem Vertriebsmitarbeiter.

- Produktlebenslaufmodell (vgl. Abschnitt 2.1.2)
  Das Modell des Produktlebenslaufs umfasst sämtliche Aktivitäten, die während des Produktlebens durchgeführt werden. Diese beginnen bei Marktforschung oder Kundenakquisition und enden bei Verschrottung oder Wiederverwendung. Die Aktivitäten sollen in einen zeitlichen Verlauf einzuordnen sein (z.B. vor, während und nach der Nutzung).

- Produktumgebungsmodell (vgl. Abschnitt 2.1.2)
  Das Modell der Produktumgebung beschreibt außerhalb der Systemgrenzen liegende Gegebenheiten, die Einfluss auf das Produkt haben. Das betrifft

reale (z.B. Nachbarsysteme, Stellflächen) und abstrakte Objekte wie z.B. logistische Prozesse (Beschickung der zu entwickelnden Maschine).

- Systemideemodell (vgl. Abschnitt 2.1.1)
  Dieses Modell stellt die verschiedenen Aspekte der Systemidee dar. Dabei berücksichtigt es ggf. verschiedene Sichtweisen, wie Herstellersicht oder Kundensicht.

- Zielmodell (vgl. Abschnitt 2.1.3)
  Das Zielsystem beschreibt die verschiedenen Ziele, die das Produkt erfüllen soll (z.B. hochdynamische Handhabung) bzw. die mit der Produktentwicklung erfüllt werden sollen (z.B. kurze Lieferzeiten). Insbesondere werden Abhängigkeiten zwischen den Zielen betrachtet, um Zielkonflikte zu identifizieren.

- Anforderungsmodell (vgl. Abschnitt 2.1.7 bis Abschnitt 2.1.9)
  Das Anforderungsmodell bildet alle notwendigen Anforderungen ab und stellt diese z.B. als Vertragsgrundlage, Entwicklungsleitfaden, Statusüberprüfung oder zum Testen bereit. Besonders wichtig ist die Darstellung von Abhängigkeiten, um Redundanzen und Widersprüche zu erkennen, die zu Zielkonflikten führen können.

- Funktionsmodell (vgl. Abschnitt 2.1.5)
  Im Funktionsmodell wird das zu entwickelnde Produkt in die zu erfüllenden Funktionen zerlegt und z.B. in Form einer Funktionsstruktur dargestellt. Dabei werden auch die Flüsse zwischen den einzelnen Funktionen berücksichtigt. Das Funktionsmodell dient als Ausgangspunkt für die weitere Entwicklung.

- Strukturmodell (vgl. Abschnitt 2.1.5)
  Das Strukturmodell stellt die strukturellen Komponenten dar. Es werden die einzelnen Module (z.B. Gelenk) und deren Anordnung modelliert. Außerdem werden die Module durch bestimmte Parameter beschrieben und Schnittstellen zu anderen Modulen und der Umgebung werden definiert.

- Verhaltensmodell (vgl. Abschnitt 2.1.5)
  Dieses Modell beschreibt das Verhalten des Produkts. Es werden Eigenschaften beschrieben, die über die reine Funktionserfüllung hinausgehen. Diese können häufig nicht formal mathematisch bzw. physikalisch beschrieben werden oder die Beschreibung ist deutlich aufwändiger als im Funktionsmodell. Beispiele sind zeitabhängige Prozesse (z.B. Scheduling der Ausführung von Prozeduren auf mehreren Prozessoren), zeitabhängige physikalische Effekte (z.B. Schwingungen bei komplexer Bauteilgeometrie und Anregungsmechanismen) oder „weiche“ Aspekte (z.B. Wertigkeit durch klapperfreie Kinematiken).

- Kostenmodell (vgl. Abschnitt 2.1.3 und Abschnitt 2.1.5)
  Im Kostenmodell werden alle interessierenden Kosten erfasst und bestimmten Kostenträgern zugewiesen. Die Kosten werden von frühen Schätzwerten ausgehend stetig konkretisiert. Dadurch ist eine entwicklungsbegleitende Kostenkalkulation möglich und die erzielbaren Selbstkosten und prognostizierten Betriebskosten können mit den zuvor festgelegten Kostenzielen verglichen werden.

- Testmodell (vgl. Abschnitt 2.1.4)
  Das Testmodell beinhaltet die Testkriterien, die für eine Bewertung und Abnahme des Produkts bzw. von Konzepten notwendig sind. Außerdem enthält es die Beschreibungen der Testfälle durch die die Erfüllung der Testkriterien überprüft wird.

Die Modelle sollen im Produktentwicklungsprozess durchgängig ein- und umgesetzt werden. Außerdem müssen Beziehungen zwischen den Partialmodellen modellierbar sein. Beispielsweise kann angegeben werden, welche Stakeholder in welchen Anwendungsfällen involviert sind. Eine partialmodellübergreifende Analyse ist notwendig, um z.B. Zielkonflikte aufzudecken. Zielkonflikte entstehen meist nicht innerhalb des Anforderungsmodells, sondern erst durch Festlegungen in konkreteren Partialmodellen.

Aus Abb. 3.3 ist ersichtlich, dass das Anforderungsmodell eine zentrale Rolle einnimmt. Es ist das einzige Modell, dass in Beziehung zu sämtlichen anderen Partialmodellen steht. Begründungen für die in der Abbildung angegebenen Beeinflussungsmöglichkeiten sind im Anhang Abschnitt B.2 zu finden.

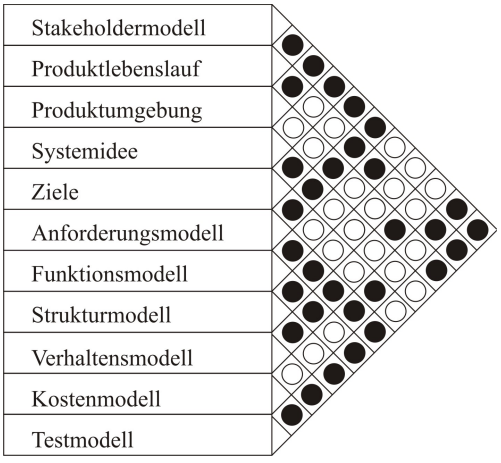


Abbildung 3.3: *Die dargestellte Beeinflussungsmatrix gibt die wichtigsten Beziehungen der Partialmodelle untereinander an (○ keine Beziehung, ● Beziehung).*

3.3 Beschreibungssprache und Werkzeug

Bereits in Abb. 2.13 in Abschnitt 2.2.6 wurde auf die Notwendigkeit der Formalisierung hingewiesen, wenn verschiedene Menschen zusammenarbeiten und über eine Gegebenheit sprechen. Diese Formalisierung erfolgt über Beschreibungssprachen, die z.B. nach *Franke* [Fra76, S. 29] oder *Ullman* [Ull92, S. 28] nach den folgenden Prinzipien unterschieden werden können:

- Semantisch (verbal): textuelle oder verbale Repräsentation eines Objekts
- Grafisch: Zeichnung eines Objekts, z.B. Skizze
- Analytisch (formal): Gleichungen, Regeln oder Prozeduren
- Physisch (körperlich): Die Hardware, das eigentliche Objekt

Ein Großteil der Beschreibungssprachen bedient sich dabei mehrerer dieser Prinzipien. Beispielsweise werden in einer Spezielle Funktionsstruktur (SFS) [Sim74] grafische und analytische Elemente verwendet. Eine Funktionsstruktur nach *Pahl und Beitz* [Pah07] bedient sich semantischer und grafischer Elemente.

Im Laufe der Jahre wurde eine große Anzahl unterschiedlicher Beschreibungssprachen z.T. zu Forschungszwecken aber auch für industrielle Anwendungen entwickelt. *Gausemeier* gibt in [Gau08a] einen kurzen Überblick über den Stand der Technik und stellt in [Gau08b] einen eigenen Ansatz für die Beschreibung selbstoptimierender Systeme vor (vgl. auch [Fra06]). Tabelle B.4 (im Anhang) gibt das Ergebnis des Stands der Forschung wieder. In der Kopfspalte finden sich siebzehn Möglichkeiten zur Beschreibung mechatronischer Systeme. In der Kopfzeile befinden sich sieben grundlegende Anforderungen, die diese Beschreibungssprachen erfüllen sollen. In der Matrix wird nun beurteilt, ob die Anforderung jeweils voll, teilweise oder nicht erfüllt wird.

Eine solche qualitative Beurteilung kann niemals objektiv erfolgen. Es lassen sich aber viel versprechende Lösungen eingrenzen. Werden als weitere Anforderungen aus den in Abschnitt 2.2 dargelegten Gründen der derzeitigen Verbreitungsgrad und die vorhandene Softwareunterstützung ergänzt, so engt sich das Feld auf die SysML ein. In den folgenden Abschnitten werden die Grundzüge der SysML und die Möglichkeiten vorhandener Softwareunterstützung vorgestellt.

3.3.1 Die Systems Modeling Language (SysML)

Die SysML ist ein gut dokumentierter Ansatz zur Modellierung auf unterschiedlichen Abstraktionsniveaus und aus unterschiedlichen Sichtweisen heraus. Zusammen mit der UML handelt es sich um eine bekannte und weit verbreitete Notation auf den Gebieten der Software- und Elektronikentwicklung, Automation und in einigen Bereichen des Maschinenbaus (z.B. Beschreibung von Flugzeugstrukturen in UML [La 06], UML-Funktionsstrukturen [Joh08], Funktions-, Verhaltens- und

Strukturmodellierung mechatronischer Systeme in SysML [Alv09]). UML und SysML werden inzwischen an vielen Universitäten gelehrt, so dass davon ausgegangen werden kann, dass die Einführungsbarrieren (z.B. Erlernen einer neuen Modellierungssprache) dieser Notation wesentlich geringer ausfallen werden als bei einer neu entwickelten akademischen Notation. Das gilt selbst dann, wenn einige zusätzliche Elemente neu definiert werden müssen, um projektspezifische Randbedingungen erfüllen zu können (vgl. Abb. 2.13).

Diese Beschreibungssprache basiert auf dem Prinzip der Objektorientierung. Die Objektorientierung stammt ursprünglich aus der Softwareprogrammierung und bildet ein natürliches menschliches Denkmuster ab, dass bereits so ähnlich von Linné für die biologische Taxonomie angewandt wurde. Der Modellierungsgegenstand wird in einzelne Objekte zerlegt, die Eigenschaften und Operationen besitzen. Eigenschaften sind zum einen eingeprägte Kennzeichen des Objekts (z.B. Farbe eines Autos), können aber auch Beziehungen zu anderen Objekten beschreiben (z.B. Enthaltungsbeziehung eines Rads zum Auto). Eine Eigenschaft besteht dabei aus einem Merkmal (z.B. Farbe) und einer Merkmalsausprägung (z.B. rot). Operationen werden auf die Objekte angewendet und beeinflussen die Objekteigenschaften. Beispielsweise setzt in Visual Basic (VB)-Code die Operation `.hide` die Ausprägung des Merkmals `.visible` eines Formularfensters auf `hidden`, so dass das Fenster verschwindet.

Ähnliche Objekte werden zu Klassen zusammengefasst. So können konkrete Objekte (z.B. Audi A6) als Instanz einer Klasse (z.B. Auto) angesehen werden. Alle Objekte der Klasse werden über die gleichen Merkmale beschrieben. Durch unterschiedliche Merkmalsausprägungen unterscheiden sie sich voneinander. Beispielsweise hat der aktuelle Audi A6 Avant ein Gepäckraumvolumen von 565 l während der Audi A4 Avant ein Gepäckraumvolumen von 490 l besitzt.

Die SysML nutzt Teile der UML und spezielle Erweiterungen zur Systemmodellierung (z.B. Anforderungsdiagramme). Ziel war es die Notation im Vergleich zur UML zu vereinfachen, indem sich auf die wesentlichen zur Systemmodellierung notwendigen Elemente beschränkt wird. Außerdem wurden Elemente ergänzt, die in der UML bisher nicht vorhanden waren, für eine Systemmodellierung aber hilfreich sind. Dabei wurde sich um eine möglichst fachbereichsunabhängige Terminologie bemüht, z.B. wurde auf den Begriff „Klasse“ verzichtet, um keine Ressentiments gegen Programmierung zu stützen. Dennoch werden Klassen in der objektorienierten Beschreibung verwendet (z.B. `block`), sie werden nur nicht explizit so bezeichnet.

Nachdem im September 2007 die *OMG SysML v1.0* zur Anwendung freigegeben wurde, folgte bereits im Dezember 2008 die im Detail verbesserte *OMG SysML v1.1* [Obj09]. Abb. 3.4 zeigt den Aufbau der SysML nach ihren Diagrammtypen. Sie kennt Verhaltens- und Strukturdiagramme (*Behavior* und *Structure Diagram*), die teilweise direkt aus der UML, z.B. das Anwendungsfalldiagramm (*Use Case Diagram*), oder modifiziert entnommen wurden, z.B. Aktivitätsdiagramm (*Activity Diagram*). Besonderheiten bilden die beiden neuen Diagramm-

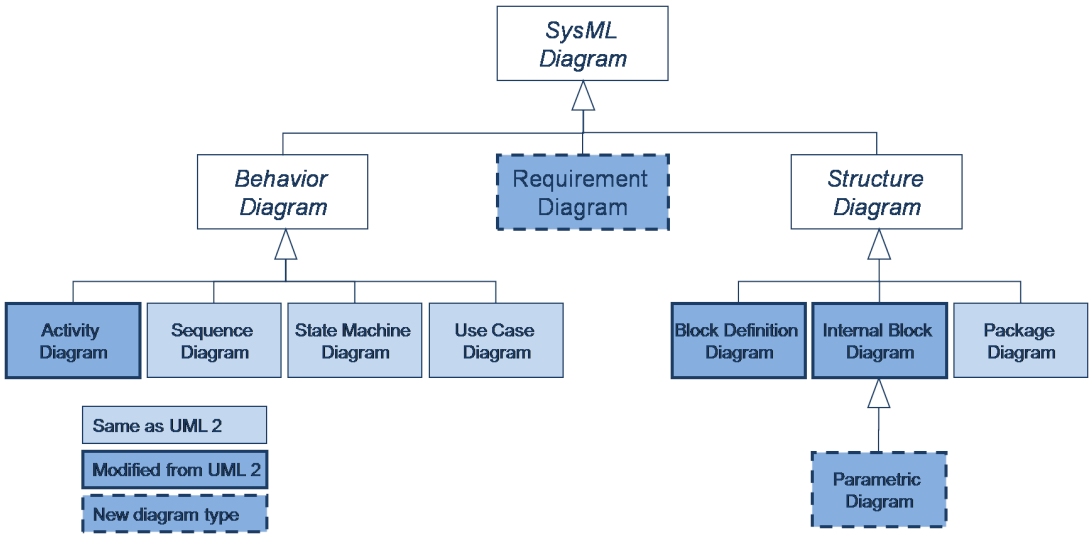


Abbildung 3.4: *Der Aufbau der SysML-Diagramme, nach [Obj09].*

typen Anforderungs- und Parametrikdiagramm (*Requirements* und *Parametric Diagram*). Durch das Anforderungsdiagramm können Anforderungen als Objekte grafisch dargestellt und mit bestimmten Relationen versehen werden, z.B. Erfüllungsbeziehungen (*Satisfy Relation*). Das Parametrikdiagramm erlaubt es, Modellparameter durch Bedingungen (*Constraints*) zu verknüpfen. Es können beispielsweise mathematische Gleichungen angegeben werden, um physikalische Abhängigkeiten quantitativ zu beschreiben.

3.3.2 SysML Entwicklungswerkzeuge

Zur Nutzung der SysML gibt es bereits einige kommerzielle und freie Software (z.B. Artisan Studio, IBM Tau), die die Spezifikation unterschiedlich aufwendig umsetzen und als Modellierungsumgebung verschiedene Funktionalitäten bereitstellen, die das Arbeiten erleichtern. Beispielsweise werden automatisch spezielle Anforderungslisten für bestimmte Sichten erstellt. Die Beziehungen werden in Matrixform (z.B. Anforderungsmatrix, Design Structure Matrix (DSM)) dargestellt und z.B. an Tabellenkalkulationsprogramme weitergegeben. Weiterhin ist eine Automatisierung und angepasste Erweiterung der Funktionalitäten durch Makros möglich. Einige bereitgestellte Schnittstellen (z.B. zu Simulink), automatische Codegenerierung (z.B. C++, Java), Versionierung und Änderungsmanagement runden die Funktionalitäten ab. Für den im Rahmen dieser Arbeit verwendeten Funktionsumfang ähneln sich die kommerziellen für die SysML-Modellierung angebotenen Werkzeuge sehr stark. Lediglich einfache—hauptsächlich für die Anzeige von Diagrammen entwickelte—Software (z.B. MS Visio) ist durch den vergleichsweise geringen Funktionsumfang hier ungeeignet. Eine Gegenüberstellung aktueller Werkzeuge findet sich z.B. im Internet [Sys10]. Die

Entscheidung zur Verwendung von Artisan Studio [Art10] beruht im wesentlichen auf dem Lizenzmodell (multiuser Hochschullizenz mit Repository, freie singleuser Lizenzen mit begrenztem Funktionsumfang) und dem Support (Webinare, Lehrunterlagen). Dadurch wurde eine flexible Verwendung in Forschung und Lehre möglich.

Das kooperative Arbeiten wird durch viele der Werkzeuge bereits unterstützt. So wird z.B. ein gemeinsames *(Multi-User) Repository* auf einem über das Netzwerk erreichbaren Server abgelegt. Der Server übernimmt die Verwaltung der Modelle und die Sicherstellung des Sicherheitsstandards (z.B. Benutzer- und Rechteverwaltung).

Die für eine Zusammenarbeit notwendige Versionierung der Modelle wird unter anderem von *Schichtel* theoretisch beschrieben [Sch02, S. 82-97]. In allen Bereichen, in denen Daten in unterschiedlichen Entwicklungsständen verwendet werden, wird mit unterschiedlichen Versionen gearbeitet. Im Allgemeinen werden Modellbäume generiert in denen das oberste Modell des Stamms von der jeweils allgemeingültigen Version—dem aktuellen Stand des Projekts—gebildet wird. Die verschiedenen Projektmitarbeiter bzw. Projektgruppen generieren Äste und arbeiten in sogenannten *Private Sandboxes* an speziellen Teilaufgaben. Die verschiedenen Versionen werden auf zwei unterschiedliche Weisen zusammengeführt. Als *rebase* bezeichnet man die Integration der aktuell gültigen Stammversion in eine Unterversion eines Astes. Die durch diese Operation entstandene neue Version ist wieder Teil des Astes. Beim *reconcile* wird dagegen die Unterversion des Astes in die Stammversion integriert. Das Ergebnis der Operation bildet das neue oberste Modell des Stamms, d.h. die neue allgemeingültige Version.

Während der Zusammenführung zweier Modelle werden die Unterschiede sichtbar, so dass z.B. erkennbar wird, wenn in einem Fachbereich Änderungen am Modell vorgenommen wurden, die sich auf weitere Teile des Modells (z.B. im Gebiet anderer Fachbereiche) auswirken. In Abb. 3.5 ist ein Beispiel für die Analyse verschiedener Modellversionen eines Parallelroboterprojekts zu sehen. Auf der linken Seite sind die von Veränderungen betroffenen Objekte der neueren und auf der rechten Seite die entsprechenden Objekte der älteren Version angegeben. Die automatisch durchgeführte Analyse ergab in diesem Fall, dass es zwischen den Versionen einige entscheidende Unterschiede gibt. Beispielsweise wurde der benötigte Freiheitsgrad der kinematischen Struktur von drei auf vier verändert und die maximale Nennlast auf den Wert 3 kg festgelegt. Diese Objekte haben entscheidenden Einfluss auf das Gesamtsystem und zahlreiche Komponenten. Hier muss im Detail untersucht werden, welche weiteren Änderungen nun ebenfalls notwendig werden. Beispielsweise müssen die Gelenke hinsichtlich der nun festgelegten Belastungen ausgewählt bzw. ausgelegt werden. Es ist daher wichtig, dass die Zusammenführung und die Analyse der Änderungsauswirkungen regelmäßig vom Projektmanagement eingefordert werden. Außerdem sollten ausreichend viele und verständliche Kommentare im Modell und insbesondere bei Änderungen gesetzt werden, damit die Gründe für Änderungen nachvollziehbar

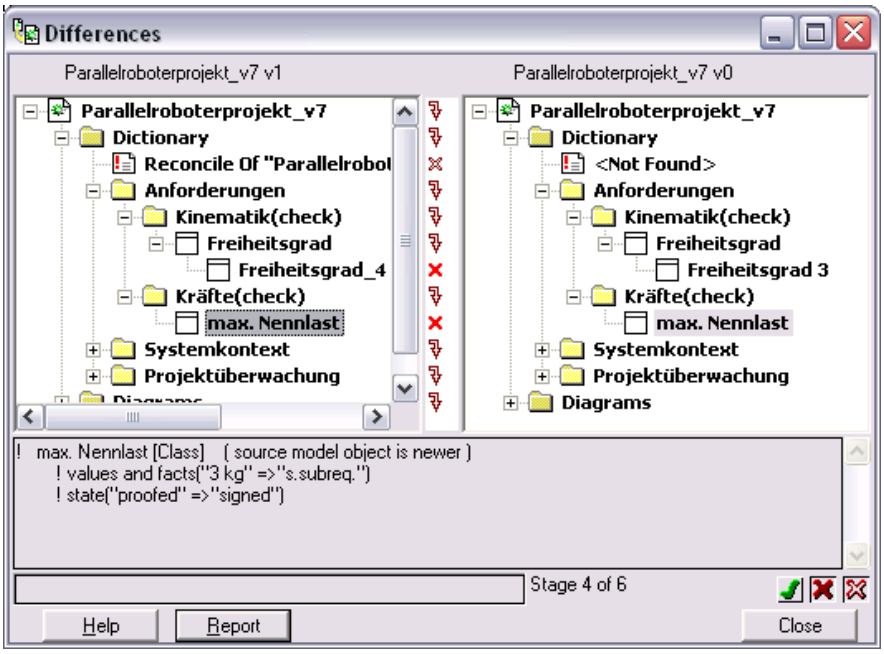


Abbildung 3.5: *Vergleich der Unterschiede in verschiedenen Versionen eines Modells für Parallelroboter.*

sind und Konflikte ggf. besser aufgelöst werden können.

Ein großer Vorteil der Versionierung ist neben der Möglichkeit der zeitgleichen Bearbeitung der Aufgabe durch mehrere Projektmitarbeiter unterschiedlicher Fachbereiche, das Untersuchen von Varianten. Ausgehend von der gleichen Stammversion können Äste mit unterschiedlichen Lösungsprinzipien weiterverfolgt und später miteinander verglichen werden. Außerdem kann die Stammversion als Muster genutzt werden, dass bereits viele gemeinsame Anforderungen erfüllt. Für spezifische Wertebereiche der Anforderungen werden dann jeweils unterschiedliche Varianten erzeugt. Damit kann die z.B. von *Jeschke* [Jes96, S. 43ff] geforderte hohe Kundenneutralität bis zum Ende der Wertschöpfungskette aufrecht erhalten bleiben und ggf. für notwendige Anpassungen vorhandene Prozesse wiederverwendet (bzw. vererbt) werden. Der Begriff der Variante unterscheidet sich dabei von dem Begriff Version. Aus einer Version können zwei unterschiedliche Folgeversionen entstehen, die sich in bestimmten Merkmalsausprägungen unterscheiden. Damit sind diese Versionen gleichzeitig Alternativen, von denen eine ausgewählt wird oder die parallel weiterverfolgt werden. In letzterem Fall bezeichnet man sie häufig als Varianten.

Größter Nachteil der beschriebenen SysML-Notation ist der starke Fokus auf die Bedürfnisse der Software- und Elektronikentwicklung (vgl. auch Tabelle B.4). Allerdings kann dieser Nachteil durch eine geeignete Methodik und eigens dafür erarbeitete Profile mit neuen Stereotypen bewältigt werden. Dazu werden neue `tag definitions` erstellt, die als Merkmale einem beliebigem vorhandenen

oder neu erzeugten `stereotyp` hinzugefügt werden können. Die Merkmalsausprägungen können z.B. als Freitext, Liste oder als Referenz zu anderen Objekten des Modells festgelegt werden. Dadurch kann z.B. einem vorhandenen Stereotyp `<<testcase>>` das neue Merkmal `aggregation level` zugefügt werden. Oder es kann ein neuer Stereotyp `<<goal>>` erzeugt werden. Die genaue Ausarbeitung einer sinnvollen Methode und zweckmäßigen Anpassung der SysML folgt in Abschnitt 4.

3.4 Auswertung

Die Auswertung der Modelle kann auf verschiedene Arten erfolgen. Die diagrammatische Auswertung findet dabei innerhalb der SysML-Modellierungsumgebung Artisan Studio 7.1 statt. Hier werden spezifische Diagramme bezüglich der konkreten Aufgabenstellung erzeugt. Bei der tabellarischen Auswertung werden die Modellinformationen aus der SysML-Modellierungsumgebung an das Tabellenkalkulationsprogramm Microsoft Excel übergeben, indem durch VB-Makros Tabellen bzw. Matrizen erzeugt werden. Dabei greift das in Excel gestartete Makro auf das zuletzt geöffnete Artisan-Modell zu und erzeugt die vorgegebene Matrix aus den Modelldaten. Anschließend kann die so erzeugte Matrix innerhalb Excel weiterverarbeitet werden.

Grundlagen der rechnerunterstützten Auswertung sind die Klassifikation der Objekte und Beziehungen, sowie das Erstellen von Kennzahlen, Diagrammen, Tabellen und Matrizen.

3.4.1 Klassifikation

Anforderungen werden klassifiziert, um eine übersichtliche Struktur zu ermöglichen. Dadurch wird selbst eine große Anzahl von Anforderungen überschaubar und durch die Entwickler handhabbar. Entwickler können bei der Bearbeitung einer konkreten Fragestellung gezielt auf die Anforderungen einer Klasse zugreifen. Insbesondere bei großen und verteilten Projekten werden in den Klassen Anforderungen unterschiedlicher Ursprünge zusammengefasst, so dass eine Bereichsübergreifende Analyse möglich ist. Beispielsweise werden alle terminlichen Anforderungen aufgelistet, egal ob sie die mechanische Struktur oder die Softwareentwicklung betreffen. Abhängigkeiten und Terminkonflikte sind schneller identifizierbar.

Die Klassifikation von Anforderungen erfolgt nach unterschiedlichen Gesichtspunkten und ist abhängig von den speziellen Erfordernissen des konkreten Projekts. Eine Klassifikation kann demzufolge nicht vollständig und allgemeingültig aufgestellt werden, sondern lediglich zweckmäßig für ein Projekt festgelegt werden. Im Folgenden werden einige aus der Literatur entnommenen Merkmale aufbereitet und strukturiert zusammengestellt. Eine zweckmäßige Festlegung oder

Tabelle 3.1: *Strukturierungsmerkmale für unternehmerische Anforderungen.*

Termin [Kru00] [Pah07] [Ehr07]
Wirtschaftlich [Fra76] [Fri90] [Bre93] [Kru00] [Pah07] [Ehr07]
Rechtlich [Rup07]
- *Normativ* [Fra76] [Koo06] [Ehr07]
- *Gesetzlich* [DIN05] [Ehr07] [Rob08]
- *Vertraglich* [Koo06]
Organisation, Management [Kru00]
- *Operationen* [Fri90] [Rup07] [Ehr07] [Rob08]
- *Rahmenbedingungen* [Koo06] [Ehr07]
Marketing, Vertrieb [Kru00]
- *Markteinführung* [Bre93]
- *sonstige Lieferbestandteile* [Rup07]
- *Nutzerklasse* [IEE98b]

Eingrenzung kann nur unter Berücksichtigung einer konkreten Aufgabe mit konkreten Randbedingungen (z.B. Branche, Unternehmensstruktur) erfolgen.

In der Softwareentwicklung wird im Wesentlichen zwischen funktionalen und nicht-funktionalen Anforderungen unterschieden, z.B. [Bun04]. *Pahl und Beitz* schlagen eine Einteilung nach sogenannten Hauptmerkmalen vor [Pah07]. Eine eindimensionale Einteilung erscheint nicht in jedem Fall ausreichend. Vielmehr ist eine Einteilung nach verschiedenen Gesichtspunkten notwendig, um die Komplexität des Gesamtsystems handhabbar zu machen. Bei *Franke* findet man z.B. in der Suchmatrix eine zweidimensionale Einteilung nach Lebenslaufphasen und Kategorien (z.B. Natur, Technik, Mensch) [Fra76, S. 136ff], wodurch sich kleinere „Teilsysteme“ ergeben.

Ein wesentlicher Einteilungsaspekt ist der Lebenslauf. Hinweise auf den Produktlebenslauf finden sich mehr oder weniger detailliert in [Fra76] [Fri90] [Bre93] [IEE98b] [Kru00] [Wei06] [Pah07] und [Rob08] , wobei lediglich *Franke* eine formal vollständige Betrachtung vorschlägt.

Ein weiterer sinnvoller Einteilungsaspekt ist der Fachbereich aus dem die Anforderung entstammt bzw. den sie bei der Entwicklung betrifft (vgl. [Bre93]). So können Anforderungsrepräsentationen für die speziellen Sichten der entsprechenden Fachbereiche aufbereitet werden.

Die Einteilung nach unternehmerischen Aspekten (vgl. Tabelle 3.1) kann nach terminlichen, wirtschaftlichen, rechtlichen, organisatorischen oder das Marketing betreffenden Gesichtspunkten erfolgen.

Technische Anforderungen (vgl. Tabelle 3.2) können in Funktion, Struktur und Verhalten unterteilt werden. Dabei enthält die Gruppe Funktion die für die Konzipierung besonders bedeutsamen funktionalen Anforderungen. Unter Struktur fallen Anforderungen, die z.B. die Geometrie oder Kinematik betreffen. Das Verhalten berücksichtigt Aspekte wie Leistung (*performance*), Zuverlässigkeit, Sicherheit und menschbezogene Anforderungen, wie Ergonomie und Ästhetik.

Tabelle 3.2: *Technische Anforderungen als Strukturierungsmerkmale.*

Funktion [Kru00] Funktion [Fri90] [Suh90] [Bre93] [IEE98b] [DIN05] [Bun04] [Wei06] [Rup07] [Rob08] [Koo06] Funktionsgrößen [Kru00] - *Energie* [Kru00] [Pah07] - *Stoff* [Kru00] [Pah07] - *Signal* [Kru00] [Pah07] Störeffekt [Kru00] - *Störkompensation und Sollwertfolge* [Lun06] - *Input-Randbedingung* [Suh90]
Struktur Geometrie [Kru00] [Pah07] Objekte [IEE98b] Kinematik [Kru00] [Pah07] Qualität [Bun04] [Rup07] Kräfte [Kru00] [Pah07]
Verhalten Zuverlässigkeit [Wei06] - *Robustheit* [Lun06] - *Stabilität* [Lun06] Performance [DIN05] [Bun04] [Lun06] [Wei06] [Rob08] - *Dynamik* [Lun06] - *Leistung* [DIN05] Menschbezogen [Fra76] - *psychologisch* [Kan84] [Bre93] - *Ergonomie, Design* [Fri90] [Kru00] [Wei06] [Rup07] [Pah07] [Rob08] - *Kulturell und Politisch* [Rob08] - *Mensch-/ Gesellsch.-/ Umwelt-Schnittst.* [Ehr07] Ökologie [Bre93] [Fri90] Sicherheit [Kru00] [Bun04] [Pah07] [Rob08]

Als letzter Aspekt ist die Systemsicht von Bedeutung. Bezieht sich die Anforderung auf das zu entwickelnde System, auf die Nachbarsysteme (technische oder menschliche) oder Schnittstellen [Suh90] [Kru00] [Ehr07]?

3.4.2 Beziehungen

Zwischen Objekten besteht eine Vielzahl unterschiedlicher Beziehungen. Werden die Beziehungen systematisch erfasst, kann durch eine Auswertung der Beziehungen auf neue Objekte geschlossen werden, d.h. die Modelle werden vervollständigt (vgl. [Hum95] [Jun06]). Außerdem kann durch eine Beziehungsanalyse das Modell auf Inkonsistenzen untersucht und so z.B. Zielkonflikte identifiziert werden. Damit eine zweckmäßige und zielführende Auswertung erfolgen kann, müssen zunächst die Beziehungen sinnvoll klassifiziert werden. Aus der Literatur sind einige Klassifikationen bekannt, von denen allerdings keine für die vorliegende Anwendung als

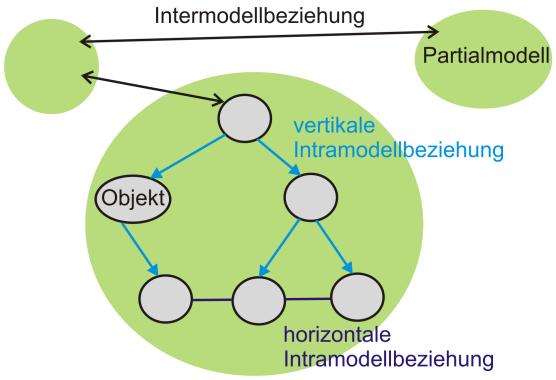


Abbildung 3.6: *Übersicht zur Unterteilung von Beziehungsarten in Bezug auf die Modellhierarchie.*

vollständig betrachtet werden kann. Im Folgenden werden die bekannten Ansätze zusammengeführt, von Widersprüchen bereinigt und sinnvoll ergänzt. Auch diese Zusammenstellung—obwohl deutlich umfassender als alle bisherigen—erhebt nicht den Anspruch einer allgemeingültigen Vollständigkeit. Im hier betrachteten Zusammenhang kann sie zumindest als hinreichend und zweckmäßig angesehen werden.

Eine in der Literatur beschriebene (z.B. [Klä93] [Hum95] [Kru00]) und zielführende Praxis ist es, Beziehungen zuerst in Inter- und Intramodellbeziehungen zu unterteilen. Die in Abb. 3.6 als schwarze Pfeile dargestellten Intermodellbeziehungen beschreiben einen Zusammenhang zwischen Elementen verschiedener (Partial-)Modelle (z.B. Anwendungsfall ↔ Anforderung). In Abb. 3.3 wurde bereits eine Matrix der in dieser Arbeit beschriebenen Partialmodelle gezeigt. Diese gibt an, zwischen welchen Partialmodellen Beziehungen bestehen. Es wird zwischen Herkunfts- und Erfüllungsbeziehungen unterschieden [Hum95, S. 97ff] [Kru00, S. 78]. Darüberhinaus existieren Äquivalenzbeziehungen [Cha08, S. 135ff], die ebenfalls den Intermodellbeziehungen zuzuordnen sind.

- Herkunftsbeziehung
  Ein Element eines Partialmodells wird durch ein Element eines weiteren Partialmodells erzeugt, z.B. kann eine Anforderung aus einem konkreten Anwendungsfall entwickelt bzw. durch diesen konkretisiert werden. Dadurch erhöht sich der Informationsgehalt im konkreteren Modell.

- Erfüllungsbeziehung
  Eine Erfüllungsbeziehung stellt eine gerichtete Relation zwischen zwei Elementen verschiedener Partialmodelle dar. Sie beschreibt, dass das eine Element einen Erfüllungsbeitrag an das andere leistet. Beispielsweise kann ein konkretes Bauteil oder eine Baugruppe (z.B. Batterie) einen Beitrag zur Erfüllung einer funktionalen Anforderung (z.B. „*mobile Energiequelle mitführen*“) leisten. Ein Spezialfall der Erfüllungsbeziehung ist die `Zweck-Mittel-`

Beziehung [Bid68, S. 105] [Kru00, S. 74].

- Äquivalenzbeziehung
  Ein Element eines Partialmodells wird in einem weiteren Partialmodell ebenfalls verwendet. Dabei kann es sich um eine Kopie (`ist-gleich`) oder ein Synonym handeln.

Die oben genannten Äquivalenzbeziehungen können ebenfalls innerhalb eines (Partial-)Modells vorkommen, sind dann allerdings oft ein Fehler. Beispielsweise erhöhen doppelt vorkommende Anforderungen (Redundanz) oder synonym verwendete Begriffe lediglich die Komplexität. Gleichzeitig erhöht sich das Risiko von Fehlentwicklungen und unentdeckten Zielkonflikten. Letztere können z.B. auftreten, wenn zwei redundante Anforderungen in unterschiedlicher Weise konkretisiert werden. Beispielsweise können sich bei einer Roboterentwicklung für die redundanten Anforderungen „*Online-Kalibrierung ermöglichen*“ und „*Roboter während des Betriebs kalibrieren*“ verschiedene Projektmitarbeiter verantwortlich fühlen. So können bereits Softwaremodule für eine Kalibrierung mit Hilfe interner Sensorik programmiert worden sein, während gleichzeitig externe Sensorik in die Gestellstruktur integriert wurde.

Die Intramodellbeziehungen werden in dieser Arbeit in vertikale und horizontale Beziehungen unterschieden. Die in Abb. 3.6 als dunkelblaue Pfeile dargestellten vertikalen Beziehungen sind hierarchisch in einer Baumform beschreibbar. Sie werden z.B. bei *Krusche* allgemein als `ist-Teil-von` bezeichnet [Kru00, S. 73ff]. *Chahadi* unterscheidet `ist-Oberbegriff` und `ist-Unterbegriff`. *Kruse* gibt eine feinere Einteilung an [Kru96, S. 78ff], die hier wie folgt weiterverwendet werden soll:

- Spezialisierung
  Ein Element wird durch den Zuwachs von Eigenschaften detaillierter beschrieben. Beispielsweise wird eine Anforderung mit Fortschreiten des Projektes erst qualitativ und später quantitativ beschrieben (z.B. *Es soll eine hohe Beschleunigung am Tool Center Point (TCP) erzielt werden* → *Es soll eine Beschleunigung von* $a_{TCP} = 10 \cdot g$ *am Tool Center Point (TCP) erzielt werden*). Das Gegenteil der Spezialisierung ist die Generalisierung.

- Dekomposition
  Durch die Dekomposition wird ein Element in Unterelemente zerlegt. Beispielsweise kann die Anforderung *Großes Arbeitsraumvolumen bereitstellen* in die Anforderungen *Große Arbeitsraumlänge bereitstellen*, *Große Arbeitsraumbreite bereitstellen* und *Große Arbeitsraumhöhe bereitstellen* zerlegt werden. Das Gegenteil der Dekomposition ist die Zusammenführung. Logische Operationen fallen zum Teil in diesen Bereich (z.B. und, oder).

- Variation
  Die Variation ergibt unterschiedliche Ausführungen des gleichen Elements.

Beispielsweise kann die Anforderung *Arbeitsraumlänge* für einen bestimmten Anwendungsfall den Wert $l_{AR} = 500\,\text{mm}$ für einen anderen Anwendungsfall jedoch den Wert $l_{AR} = 1000\,\text{mm}$ annehmen. Die Variation kann dabei mit einer Versionierung einhergehen.

Die in Abb. 3.6 als hellblaue Linien dargestellten horizontalen Beziehungen werden danach charakterisiert, auf welche Weise sie Objekte in Beziehung zueinander setzen (vgl. auch [Ste08b] [Ste09a]). Dabei muss eine Beziehung durch mehrere Merkmale beschrieben werden. Beispielsweise kann eine *quantitativ* beschriebene Beziehung sich *negativ* auswirken. Falls zwischen zwei Objekten keine Beziehungen existieren gelten sie als unabhängig voneinander, d.h. sie haben keinen gegenseitigen Einfluss. Im Folgenden werden die Beziehungsmerkmale und deren Merkmalsausprägungen erklärt.

- Granularität
  Die Granularität beschreibt den Detaillierungsgrad der in Beziehung gesetzten Objekte [Ari07]. Ein System kann eine aus Bauteilen zusammengesetzte Baugruppe sein. Auf abstrakter Ebene ist ein System eine Gruppe von beliebigen abstrakten Elementen, z.B. Anforderungen, Funktionen, Anwendungsfälle. Das betrachtete System ist dann z.B. eine Anforderungsliste, eine Funktionsstruktur oder ein Anwendungsfalldiagramm (*use-case diagram*). Die Elemente eines Systems können selbst wieder in Elemente zerlegt werden. Eine Beziehung kann Elemente gleicher oder unterschiedlicher Ebenen verbinden.

  - Auf gleicher Ebene
    Diese Beziehung besteht zwischen zwei Elementen auf der gleichen granularen Ebene, z.B. zwischen zwei Anforderungen mit gleich hoher Spezialisierung.
  - Auf unterschiedlicher Ebene
    Beziehungen zwischen Elementen auf unterschiedlichen granularen Ebenen [Kru00, S. 75], z.B. die Anforderungen *Großen Arbeitsraum bereitstellen* und *Überstreichbaren Gelenkwinkel von* $\alpha \geq 275^o$ *realisieren.*

- Quantifizierbarkeit
  Beziehungen lassen sich insbesondere in frühen Entwicklungsphasen häufig nur qualitativ angeben. Im weiteren Verlauf des Produktentwicklungsprozesses sind bei größerer Spezialisierung und Konkretisierung häufig quantitative Aussagen möglich.

  - Quantitativ
    Eine quantitative Beziehung wird mathematisch z.B. auf Basis physikalischer oder wirtschaftstheoretischer Grundlagen beschrieben [Hum95, S. 85ff] [Kru00, S. 73ff]. Dabei kann es sich um eine Gleichung oder

ein Gleichungssystem handeln. In der Regel können quantitative Beziehungen erst dann festgelegt werden, wenn ein gewisser Konkretisierungs- und Spezialisierungsgrad erreicht ist. Beispielsweise können quantitative Beziehungen zwischen funktionalen Anforderungen gut beschrieben werden, wenn eine spezielle Funktionsstruktur [Sim74] bereits erstellt wurde, d.h. verbindende physikalische Effekte vorliegen.

- Qualitativ
  Qualitative Beziehungen sind nicht durch physikalische o.ä. Zusammenhänge mathematisch beschreibbar [Klä93, S. 168] [Hum95, S. 85ff] [Kru00, S. 73ff]. Gründe hierfür sind z.B. eine hohe Komplexität der Zusammenhänge, für die keine sinnvolle mathematische Repräsentation gefunden werden kann. Die Aussage, dass ein Zusammenhang besteht, resultiert dabei aus Expertenwissen. Qualitative Aussagen auf Basis von Expertenwissen sind dabei kritisch zu betrachten. Experten liegen bei typischen Anwendungen aus ihrem Wissensgebiet sehr häufig richtig, bei neuen Lösungen dagegen oft falsch. Hier trügt die Erfahrung, falls sie auf neue Technologien nicht übertragbar ist.
- Quantifizierbar
  Beziehungen, die auf Grund des geringen Konkretisierungsgrades bisher nur qualitativ vorliegen, werden in späteren Phasen ggf. quantitativ beschrieben [Kru00, S. 94]. Beispielsweise existiert immer eine Beziehung zwischen der Masse und der Steifigkeit eines Bauteils. Diese kann jedoch erst in späteren Phasen quantitativ angegeben werden. Beispielsweise ergibt sich die Steifigkeit bzgl. Durchbiegung eines einseitig eingespannten Rechteckträger mit Vollquerschnitt und einer Belastung durch eine Punktlast zu

$$\delta = \frac{F}{f} = \frac{E}{4} \cdot \frac{b \cdot h^3}{l^3} \tag{3.1}$$

$$m = \rho \cdot l \cdot b \cdot h \tag{3.2}$$

  mit δ-Biegesteifigkeit, F-Punktlast am Balkenende, f-Durchbiegung am Balkenende, E-Elastizitätsmodul, b-Breite, h-Höhe, m-Masse, ρ-Dichte und l-Länge des Rechteckträgers.

- Richtung
  Beziehungen zwischen Elementen sind häufig gerichtet, d.h. sie haben einen Start- und einen Endpunkt. Der Fluss kann in einer festgelegten Richtung oder wechselseitig möglich sein.

  - unidirektional
    Bei zwei unidirektional miteinander verbundenen Elementen wird ein Element von dem anderen beeinflusst [Kru00, S. 73ff]. Gleichzeitig

wirkt die Beziehung nicht auf das beeinflussende Element zurück. Beispielsweise sendet ein Türöffner ein Signal, welches zum Öffnen der Tür führt. Der Türöffner selbst wird über diese Beziehung im Allgemeinen von der Tür nicht beeinflusst.

– wechselseitig
Zwei wechselseitig miteinander verbundene Elemente beeinflussen sich gegenseitig [Kru00, S. 73ff]. Das bedeutet die Verfolgungsrichtung von einem Element zum anderen ist umkehrbar. Beispielsweise sind Funkgeräte gleichzeitig als Sender und Empfänger ausgeführt.

- Stärke
Häufig kann eine Stärke der Beziehung angegeben werden. Im Idealfall einer als mathematische Funktion quantifizierten Beziehung wird ein Zahlenwert angegeben. Werden beispielsweise alle Parameter außer h der Gl. (3.1) und Gl. (3.2) festgelegt, so ergibt sich bei Variation von h der in Abb. 3.7 graphisch dargestellte Zusammenhang. Dabei nimmt das Gewicht des Rechteckträgers mit zunehmender Biegesteifigkeit zunächst überproportional zu. Im weiteren Verlauf ist mit einer vergleichsweise geringen Massezunahme eine große Steifigkeitserhöhung zu erreichen. Die Empfindlichkeit lässt sich in diesem speziellen Fall durch die partielle Ableitung

$$\frac{\partial m}{\partial \delta} = \frac{4^{1/3}}{3} \cdot \frac{\rho \cdot l^2 \cdot b^{2/3}}{\sqrt[3]{E \cdot \delta^2}} \tag{3.3}$$

beschreiben. So kann im konkreten Fall eine Entscheidung darüber getroffen werden, ob eine Optimierung des Parameters sinnvoll ist (bei geringer Empfindlichkeit), oder eine vollständig neue Lösung erdacht werden muss

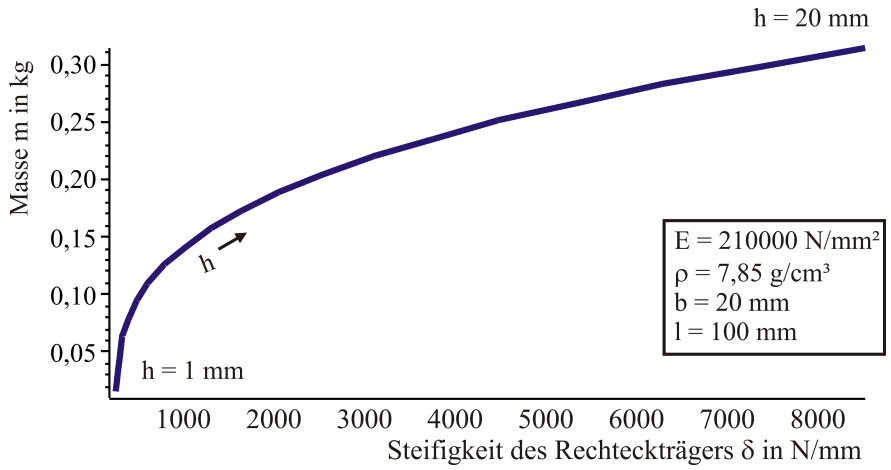


Abbildung 3.7: *Zusammenhang zwischen Biegesteifigkeit und Masse eines Rechteckträgers mit Vollquerschnitt bei Variation der Höhe h.*

(z.B. Verwenden von Hohlprofilen).
Oftmals kann die Stärke der Beeinflussung jedoch nur qualitativ, z.B. in einer Abstufung *schwach, mittel, stark* oder prozentual, angegeben werden.

- Unterstützung
  Kühnpast untersucht für die Gestaltung maschinenbaulicher Konstruktionen das Prinzip der Selbsthilfe. Eine selbsthelfende Konstruktion (z.B. Radialwellendichtring) liegt dann vor, falls sich Ursprungswirkung (Dichtkraft aus Ringfeder) und Hilfswirkung (Dichtkraft aus höherem Innendruck) additiv zu einer Gesamtwirkung zusammensetzen. Darüberhinaus lassen sich selbstschadende, selbsthilfeneutrale und in Sonderfällen selbsthemmende Lösungen unterscheiden. [Küh68, S. 17ff]
  Für die Betrachtung der Beziehungen auf abstrakter Modellebene wird in dieser Arbeit der Einfluss den zwei Elemente aufeinander ausüben als unterstützend, konfliktär oder ausschließend aufgefasst. Dieses Merkmal entscheidet darüber, ob ein Zielkonflikt vorliegt. Da Objekte selten rein dipolar in Beziehung stehen, ist im konkreten Fall jedoch abzuwägen, welche Beziehung den entscheidenden Einfluss ausübt.

  - positiv
    Eine positive Beziehung bedeutet, dass bei einem besseren Wert des einen Elements gleichzeitig ein besserer Wert des anderen Elements folgt. Wird beispielsweise ein größerer Roboterarbeitsraum gefordert, ist im Allgemeinen eine bessere Zugänglichkeit für Wartungstätigkeiten zu erwarten. Die Beziehung wirkt unterstützend [Kru96, S. 81] [Kru00, S. 73ff].

  - negativ
    Eine negative Beziehung löst in vielen Fällen einen Zielkonflikt aus [Klä93]. Bei einem besseren Wert des einen Elements folgt zwangsläufig ein schlechterer Wert des anderen Elements [Kru96, S. 81] [Kru00, S. 73ff]. Hier müssen Konfliktlösungsstrategien angewendet werden, um den Zielkonflikt zu lösen (Konsens) oder zumindest zu entschärfen (Kompromiss). Konfliktlösungsstrategien werden später in diesem Abschnitt noch genauer untersucht.

  - ausschließend
    Die ausschließende Beziehung bildet den Grenzfall der negativen Beziehung. Hier kann nur eines der beiden Elemente gleichzeitig erfüllt werden. [Kru96, S. 81] [IEE98b, S. 6] [Kru00, S. 73ff]

- Verbindung
  Beziehungen verbinden zwei Elemente. Dabei wird zwischen direkter (unmittelbar) oder indirekter (mittelbar) Verbindung unterschieden.

  - direkt
    Eine direkte Verbindung zwischen zwei Elementen liegt dann vor, falls die Beeinflussung unmittelbar ist. Beispielsweise besteht eine direkte Verbindung zwischen dem Absatzgebiet eines Produktes und dem zu berücksichtigenden Temperaturbereich. In Äquatornähe sind deutlich höhere Temperaturen zu erwarten als in Nordeuropa oder Kanada. Folglich müssen z.B. entsprechende Schmiermittel ausgewählt werden.
  - indirekt
    Eine indirekte Verbindung ist durch das Vorhandensein eines dritten Elements charakterisiert, welches die indirekte Verbindung herstellt. Beispielsweise hat die maschinelle Umgebung (z.B. Fördermittel) einen indirekten Einfluss auf die Zugänglichkeit bei Wartung einer Handhabungsmaschine. Läuft ein Fließband durch den Arbeitsraum, so ist er vom Wartungstechniker nur noch von maximal zwei Seiten zugänglich. Häufig bilden indirekte, negative Beziehungen den Ausgangspunkt schwer zu identifizierender Zielkonflikte. Andererseits können bestehende Zielkonflikte aufgelöst werden, indem eine direkte Beziehung in eine indirekte umgewandelt wird. Die Variation eines Zwischenelements (z.B. als Anpassbaustein [Pah07, S. 665]) zieht im Allgemeinen deutlich weniger Änderungen am Gesamtsystem nach sich.

- Umstand
  Die Verbindungstypen des Umstands (vgl. [Hob88, S. 368ff]) sind für die Nachvollziehbarkeit von Entscheidungsprozessen von großer Bedeutung. Finale (Zweck), kausale (Grund), konsekutive (Folge), konzessive (Gegensatz) und modale (Art und Weise) Zusammenhänge sind entweder bereits durch andere Merkmale beschrieben (z.B. Richtung und Unterstützung) oder die genaue Unterscheidung ist für die Auswertung auf dieser Abstraktionsebene nicht von Bedeutung (z.B. „*wenn Stahl verwendet wird dann Lackieren*“, „*Lackieren, weil Stahl verwendet wird*“).

  - konditional (Bedingung)
    Beschreibt den Umstand, dass ein Element nur auf Grund eines zweiten existiert. Beispielsweise `wenn` das Teil im Sandgießverfahren gefertigt werden soll, `dann` müssen große Radien vorgesehen werden. Falls der erste Satzteil nicht gilt (z.B. weil es doch ein Schweißteil sein soll), so gilt auch der zweite nicht.
  - lokal (Ort)
    Die Elemente sind örtlich abhängig, d.h. die Beziehung besteht nur aufgrund eines lokalen Zusammenhangs. Werden die Elemente verlagert, so ist dieser Zusammenhang nicht mehr vorhanden. Wird beispielsweise der Motor eines Fahrzeugs im Heck angesiedelt hat er keinen

direkten Einfluss auf den beim Frontcrash verbleibenden Fußraum. Dafür bewirkt die Massenträgheit des Motors nun Kräfte von hinten in Richtung des Fahrgastraums.

– temporal (Zeit)
Der Zusammenhang ist zeitabhängig, d.h. er besteht nicht zu jedem Zeitpunkt. Beispielsweise verursacht bei einem Pkw die Beziehung zwischen „*Fahrbahn Anpressdruck erzeugen*“ und „*aerodynamischen Widerstand minimieren*“ bei Realisierung durch einen Heckspoiler in vielen Betriebssituationen einen Zielkonflikt. Bei einer gewollten Verzögerung des Fahrzeugs (bremsen) ist hier hingegen kein Zielkonflikt vorhanden. Im Gegenteil: Ein hoher Widerstand unterstützt den Bremsvorgang.

Die SysML kennt einige unterschiedliche Beziehungstypen, die im Programm als Stereotypen implementiert sind (`aggregation`, `allocate`, `association`, `composition`, `connection`, `copy`, `dependency`, `derive`, `extend`, `generalization`, `include`, `parent-child`, `refine`, `satisfy`, `trace`, `verify`). Einige davon sind auf bestimmte Objektklassen beschränkt, d.h. sind nur innerhalb eines Partialmodells zu nutzen. Die SysML-Beziehungstypen werden als Grundlage genutzt, um die oben beschriebenen Zusammenhänge abzubilden. Der konkrete Einsatz wird in Abschnitt 4.1 genauer beschrieben.

Durch die vorangegangenen Beschreibungen wurde deutlich, dass eine Festlegung der Beziehungen und eine Beurteilung darüber, ob ein Zielkonflikt vorliegt oftmals nur bei höherer Konkretisierung möglich ist. Wenn Anforderungen auf eine Realisierung bezogen werden, dann können die Beziehung genauer erfasst werden. Je weiter die Konkretisierung und Spezialisierung voranschreitet, desto sicherer sind die Aussagen. Beispielsweise entsteht zwischen den Zielen eines Fahrrads „*Anfahren mit geringem Kraftaufwand ermöglichen*“ und „*hohe Endgeschwindigkeit erreichen*“ nur dann ein Zielkonflikt, falls konventionelle Antriebskonzepte (der Fahrer tritt in die Pedale) verwendet werden und bestimmte Randbedingungen (z.B. Fahrrad ohne Gangschaltung) eingehalten werden müssen. Ein Fahrrad mit Hilfsantrieb würde zumindest diesen Zielkonflikt nicht hervorrufen.

Nach *Franke* lassen sich Widersprüche folgendermaßen einteilen [Fra76, S. 98]:

- logisch-formal,
- physikalisch,
- technisch,
- technologisch und
- wirtschaftlich.

Sind Widersprüche gefunden können sie auf Ebene der Funktionsstruktur durch folgende Variationsoperationen gelöst werden [Fra09, S. 5.7]:

- Wandler oder Leiter einfügen,
- Reihenfolge der Schaltelemente vertauschen,
- Mehrere Elemente zusammenfassen (Funktionsintegration),
- Elemente weiter differenzieren (Funktionstrennung),
- Anzahl gleicher Elemente erhöhen oder verringern,
- Systemgrenze verschieben,
- Rückführschleifen einfügen oder entfernen,
- Informationsflüsse energetisch verstärken und
- Kombinieren von Varianten.

Auf Ebene der Gestalt können Widersprüche gelöst werden durch die Variation von [Fra09, S. 6.11f]:

- Art bzw. Form (z.B. Rillenkugellager durch Schrägkugellager ersetzen),
- Abmessung bzw. Größe (z.B. größeres Lager verwenden),
- Lage bzw. Anordnung (z.B. Welle breiter abstützen),
- Anzahl (z.B. Lageranzahl erhöhen (Schrägkugellagerpakete)),
- Freiheitsgrad (z.B. Lager adaptiv einstellbar realisieren) oder
- Kopplungsart (z.B. Wälzlager durch Gleitlager ersetzen).

Um Zielkonflikte im Produktentwicklungsprozess zu lösen, ist es sinnvoll erprobte Strategien anzuwenden. *Rupp* spricht von folgenden Konsolidierungstechniken [Rup07, S. 320ff]:

- Einigung,
- Ober sticht Unter,
- Abstimmung,
- Geringstes Übel,
- Kompromiss und

- Konfigurierbarkeit.

Diese beziehen sich zum Teil auf Konfliktlösungen, die auf organisatorischer oder zwischenmenschlicher Basis stattfinden. Beispielsweise wenn Uneinigkeit zwischen Auftraggeber und Auftragnehmer darüber besteht, ob bzw. in welchem Umfang eine Funktionalität umgesetzt werden soll oder nicht. Technische Konflikte hingegen können nicht durch Abstimmung oder Einigung gelöst werden.

Häufigstes Mittel zum Lösen von technischen Zielkonflikten ist der Kompromiss. Beispielsweise werden Strukturen so leicht gebaut, dass sie gerade noch die geforderte Steifigkeit erfüllen. Durch den Kompromiss erfüllen sie jedoch kein Ziel besonders gut. Grund hierfür ist, dass es oft eine Vielzahl gleichzeitig zu erfüllender widersprüchlicher Ziele gibt. Diese können in Form von Zielfunktionsvektoren eine mehrdimensionale Paretofront aufspannen (vgl. [Kre06b] [Dei07]). Sämtliche Lösungen auf der Paretofront werden als gleich gut angesehen. Hier kann nur durch eine Gewichtung eine Lösung ausgewählt werden. Die gewählte Lösung bildet dabei stets einen Kompromiss.

Durch Konfiguration kann hingegen ein Zielkonflikt aufgelöst werden. Die Lösung ist abhängig von der Art des Zielkonflikts. Beispielsweise kann ein temporal vorliegender Zielkonflikt durch quasistatische Adaptivität gelöst werden. Dabei werden Parameter im Betrieb geändert, so dass in unterschiedlichen Betriebszuständen unterschiedliche Eigenschaften erreicht werden (z.B. quasistatische Spieleinstellung von Robotergelenken [Fra05d] [Pav06] [Ste06a] [Ste07b] [Pav08]).

Eine weitere Strategie zur Auflösung von Zielkonflikten bilden die sogenannten 40 innovativen Grundprinzipe der Theorie des erfinderischen Problemlösens (TRIZ) (z.B. [Alt84]). Können die als konfliktär identifizierten Elemente jeweils einem der 39 Parameter zugeordnet werden, so sind der Widerspruchsmatrix diejenigen „innovativen Grundprinzipe“ zu entnehmen, die zu einer Lösung des Widerspruchs führen können. Die Übertragung des Problems auf die Parameter und die geeignete Umsetzung der abstrakten Grundprinzipe in der konkreten technischen Anwendung ist jedoch häufig nur durch TRIZ-erfahrene Entwickler möglich.

Nach *Eiletz* können Zielkonflikte sehr häufig nicht getrennt voneinander gelöst werden. Vielmehr müssen alle Zielkonflikte ganzheitlich untersucht werden, da einzelne Zielkonflikte aus anderen resultieren oder die Konfliktlösung einen anderen Konflikt verschärfen kann. Dazu ist eine einfache und übersichtliche Darstellung für alle Projektbeteiligten notwendig [Eil99, S. 33f, 103]. Nur so können die positiven Mechanismen (s. Abb. 3.8 rechts) genutzt und die negativen (s. Abb. 3.8 links) vermieden werden. Zur Lösung der Konflikte sollen die folgenden Strategien verwendet werden [Eil99, S. 105]:

- Auf Änderungen des Systemumfeldes hinwirken,
- Grenzen des Zielsystems ändern,
- Zielinhalt bzw. -ausmaß ändern,

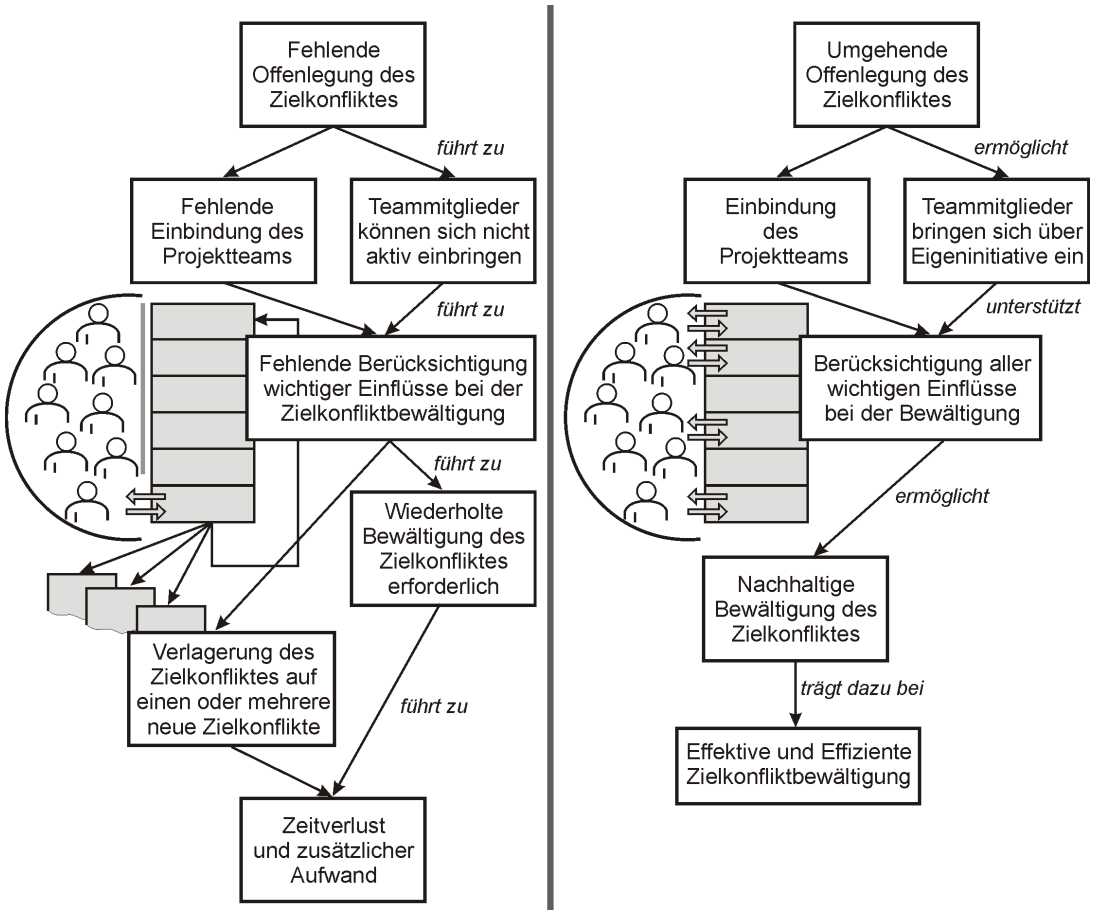


Abbildung 3.8: *Positive und negative Mechanismen beim Umgang mit Zielkonflikten, nach [Eil99, S. 102].*

- Zielbezugsbereich ändern,
- Vorhandenes Konzept optimieren,
- Neues, innovatives Konzept suchen oder
- Feld der Konzeptsuche ändern.

3.4.3 Kennzahlen

Um die Qualität der Anforderungen bzw. Spezifikation messbar zu machen werden in [Rup07, S. 358ff] Kennzahlen, genannt Metriken, eingeführt und in [Pik06] ergänzt. Allerdings wird darauf hingewiesen, dass im konkreten Projekt und Unternehmen erst nach mehreren Anwendungen eine gewisse Grundlage an Referenzdaten vorhanden sei, so dass erst dann eine sichere Aussage über die Qualität möglich ist. Eine zweckmäßige Anwendung liegt also in der Überwachung eines

Projektes im zeitlichen Verlauf. Die Beurteilung anhand eines einzelnen Wertes ist hingegen nicht sinnvoll.

Auch *Drebing* weist darauf hin, dass durch die Betrachtung von Metriken die Qualität einer Anforderungsliste beurteilt werden kann. So sind beispielsweise nur nominal- und absolutskalierte Festforderungen als Bewertungskriterien für eine Vorauswahl technischer Lösungen geeignet [Dre93, S. 120f]. Erfüllen keine oder nur wenige Anforderungen diese Bedingungen, ist eine sinnvolle Vorauswahl nicht möglich.

Metriken werden in vier Klassen eingeteilt [Rup07, S. 360] und in [Pik06] zum Teil genauer angegeben. Die erste Klasse—*textbasierte Metriken*—betrachtet im Wesentlichen die Anforderungsformulierung, z.B. die Lesbarkeit und Eindeutigkeit. Eine Anforderung wird hier als eindeutig verstanden, falls sie nicht durch Universalquantoren (und, oder) zusammengesetzt ist, d.h. es sich um eine Elementaranforderung handelt.

Strukturbasierte Metriken geben z.B. an, ob durch Vergabe einer Identifikationsnummer die Anforderungen eindeutig identifizierbar sind. *Inhaltsbasierte Metriken* geben an, wie viele Anforderungen formal vollständig sind (vgl. Gl. (3.4)).

$$1 - \frac{\Sigma \text{ unvollständige Anforderungen}}{\Sigma \text{ Anforderungen}} \tag{3.4}$$

Anforderungen werden dann als formal unvollständig betrachtet, falls nicht alle notwendigen Merkmale (z.B. Priorisierung) festgelegt wurden. Formal vollständige Anforderungen können dennoch schlecht formuliert oder mit falschen Merkmalsausprägungen versehen sein. Werden jedoch organisatorische Kontrollen eingeführt (z.B. Freigabe durch einen zweiten Entwickler), so kann eine solche Metrik bei periodischer Überprüfung zumindest dazu genutzt werden, den Projektfortschritt zu beurteilen.

Darüberhinaus werden hier Metriken zur Beurteilung der Redundanz und Notwendigkeit von Anforderungen eingeführt. Allerdings ist die Notwendigkeit nur selten mit Bestimmtheit zu beurteilen. Redundante Anforderungen sollen aus dem Anforderungskollektiv entfernt werden, sobald sie entdeckt werden. Eine Metrik, die den Anteil an redundanten Anforderungen beschreibt, wäre nur in sofern sinnvoll, als das ein hoher Anteil an Redundanzen auf Probleme in der Anforderungserfassung hindeutet. Für die Beurteilung der Qualität des aktuellen Anforderungsmodells ist sie jedoch nicht sinnvoll einzusetzen.

Als weitere Metrik wird die Modifizierbarkeit als *zusammengesetzte Metrik* vorgeschlagen. Diese wird additiv aus den jeweils mit (scheinbar willkürlich gewählten) Gewichtungsfaktoren versehenen Metriken Identifizierbarkeit, Eindeutigkeit, Redundanzfreiheit und Sortierbarkeit gebildet. Eine plausible Begründung für die Aussagekraft bleiben die Autoren schuldig.

Der IEEE-Standard 830 gibt weitere Hinweise auf mögliche Metriken [IEE98b, S. 7f]. Die Stabilität soll z.B. durch die erwarteten Änderungen pro Anforderung angegeben werden. Als Grundlage zur Schätzung soll die Erfahrung dienen. Die

generelle Gefahr erfahrungsbasierter Metriken besteht jedoch darin, dass „weiche“, ungesicherte Merkmale zu „harten“ Fakten formalisiert werden. Ein unkritischer Umgang mit solchen Metriken kann zu falschen Entscheidungen führen.

Weiterhin dient die Verifizierbarkeit zur Beurteilung der Qualität des Anforderungskollektivs. Jede einzelne Anforderung soll verifizierbar sein. Beispielsweise ist die Anforderung „*Das Auto soll eine Maximalgeschwindigkeit von mindestens* 200 km/h *erreichen*“ im Prinzip verifizierbar. „*Das Auto soll schnell sein*“ ist dagegen mangels quantitativer Werte nicht verifizierbar.

Von *Kruse* werden Kennzahlen eingeführt, die Beziehungen zwischen den Anforderungen als die Qualität beschreibende Kennzeichen auffassen [Kru96]. Sie sollen einem Projektleiter die Beurteilung der Anforderungsaufbereitungsqualität

$$A_{AQ} = \frac{\sum Beziehung}{\sum Anforderung} \tag{3.5}$$

und der Anforderungsbeziehungsqualität

$$A_{BQ} = \frac{x \cdot \sum B_{Ausschluss} + y \cdot \sum B_{Konkurrenz} + z \cdot \sum B_{Unterstützung}}{\sum Beziehung} \tag{3.6}$$

erleichtern. Dabei soll die Anforderungsaufbereitungsqualität Werte von mindestens $A_{AQ} = 0,8$ annehmen. Die Angabe eines genauen Zahlenwerts erscheint jedoch praktisch nicht sinnvoll. Wird eine einzelne Anforderung mit 80% der übrigen Anforderungen in Beziehung gesetzt ergibt sich der gleiche Wert wie für eine Vernetzung von 20% der Anforderungen mit jeweils vier der übrigen. Die Art der Vernetzung unterscheidet sich jedoch deutlich. Welches Beziehungsnetzwerk als qualitativ besser bezeichnet werden kann, hängt stark von der betrachteten Aufgabe ab.

Der Wert der Anforderungsbeziehungsqualität gibt an, welche Beziehungsart vorherrscht (z.B. $A_{BQ} \approx x$ bedeutet, dass der Großteil der Beziehungen ausschließenden Charakter besitzen). Dabei soll eine große Anzahl von ausschließenden und konkurrierenden Beziehungen auf ein hohes Entwicklungsrisiko hindeuten. Eine hohe Anzahl von unterstützenden Beziehungen soll zwar ein geringes Entwicklungsrisiko anzeigen, jedoch auch eine Anfälligkeit bei späteren Änderungen vorhersagen. Es ist jedoch fraglich, ob eine Zusammenfassung in eine Kennzahl hier eine bessere Übersicht ermöglicht. Eine einfache Gegenüberstellung (z.B. 5% Ausschluss, 25% Konkurrenz, 70% Unterstützung) scheint hier bessere Dienste zu leisten. Dennoch sind auch diese Werte nicht für eine absolute Messung der Qualität geeignet. Beispielsweise ist ein hoher Anteil ausschließender Beziehungen in einem modularen Baukastensystem unter Umständen gewollt, z.B. falls einzelne Module nicht gleichzeitig in der selben Konfiguration verwendet werden sollen.

3.4.4 Diagramme

Diagramme sind sinnvoll als Kommunikationsmedium einzusetzen (vgl. auch Abschnitt 2.2.5 und Abschnitt 2.2.6). Ausschnitte aus Partialmodellen werden mit den jeweils relevanten Elementen und Intramodellbeziehungen dargestellt und diskutiert. Weiterhin werden partialmodellübergreifende Diagramme erzeugt, die Intermodellbeziehungen abbilden. So wird es im konkreten Fall möglich, über Abteilungs- und Fachbereichsgrenzen hinweg einen ganzheitlichen Blick auf das Produkt zu bewahren. Gegenseitige Beeinflussungen werden früher gefunden und fachbereichsübergreifende Zielkonflikte können frühzeitig entschärft werden.

Über den Nutzen als reines Kommunikationsmedium hinaus geht der Einsatz von Diagrammen als Entwicklungswerkzeug. Hier werden z.B. Objekte kreiert und zu Gruppen zusammengeschoben, die eine Gemeinsamkeit aufweisen. Beispielsweise können so Module intuitiv gegeneinander abgegrenzt werden. Es werden Beziehungen zwischen den Objekten visualisiert, um z.B. Schnittstellen zwischen Modulen darzustellen. Hier werden bereits grob Packageüberlegungen visualisiert (vgl. Module Interface Graph (MIG) [Ble08a]).

Die Visualisierung der Beziehungen gestattet es, Zusammenhänge leichter zu erfassen. Der Überblick erlaubt es intuitiv weitere Zusammenhänge zu erkennen, die ebenfalls im Diagramm dargestellt werden können. Im rechnerunterstützten System werden ausgehend von einem Objekt sämtliche mit diesem in Beziehung stehende Objekte veröffentlicht. Für diese werden wieder die „Nachfolger“ veröffentlicht bis ein Netz sämtlicher Abhängigkeiten ausgehend vom Ausgangsobjekt visualisiert ist. Hier werden dann Untersuchungen z.B. zum Einfluss von Änderungen auf das Gesamtsystem unternommen.

Unter anderem werden in [Rup07, S. 413f] Verbindungstopologien nach ihrer Komplexität eingeteilt. Die Komplexität der Verbindungen nimmt dabei von linearen Ketten zu Netzen hin stark zu. Daher empfiehlt *Rupp* Netze nur dann zu verwenden, wenn es sich nicht vermeiden lässt. Die Zusammenhänge technischer Problemstellungen der Produktentwicklung sind allerdings nahezu immer so komplex, dass sie nur durch Netze sinnvoll abgebildet werden können. Sie sind daher nur sehr selten zu vermeiden. Dennoch muss stets ein Kompromiss zwischen Vollständigkeit und Verstehbarkeit gefunden werden. Dazu muss mit jedem Diagramm ein bestimmtes Ziel—eine bestimmte Sicht—verfolgt werden. Die Objekte müssen soweit abstrahiert und irrelevante Objekte vernachlässigt werden, dass die zu transportierende Information eindeutig dargestellt wird. Die Rechnerunterstützung erlaubt hier aus dem rechnerintern stark vernetzten Modell Diagramme zu erzeugen, die nur die für das betrachtete Problem notwendigen Beziehungen abbildet.

3.4.5 Tabellen und Matrizen

Je nach Zielrichtung gibt es verschiedene Möglichkeiten der tabellarischen Auswertung. Die bekannteste tabellarische Darstellung ist sicherlich die Anforderungsliste, d.h. eine nach Gliederungspunkten geordnete Auflistung der Anforderungen mit Bezeichnung, Werten, Priorisierung und Quellenangaben. Durch konsequente Verwendung eines Anforderungsmodells werden hier im Gegensatz zu statischen Anforderungslisten jeweils sichtenspezifische Listen erzeugt. Diese bilden z.B. diejenigen Anforderungen ab, die in Zusammenhang mit einer bestimmten Komponente stehen (z.B. Antrieb), zu einem bestimmten Gliederungspunkt zugeordnet werden (z.B. Energie) oder eine bestimmte Merkmalsausprägung besitzen (z.B. `revise`, vgl. Abschnitt 4.1.1). Außerdem werden—je nach Bedarf—die Anforderungsmerkmale (z.B. Priorität, Status) in der Liste abgebildet. Darüberhinaus können die unterschiedlichen Beziehungen zu den verschiedenen Objekten ebenfalls in der Liste berücksichtigt werden.

Beziehungen zwischen Objekten lassen sich außerdem übersichtlich in Form von Matrizen darstellen (z.B. DSM). Dabei werden Objekte eines Partialmodells in den Kopfspalten und Objekte eines weiteren Partialmodells in den Kopfzeilen angeordnet. Die Objekte der Kopfspalten und -zeilen werden abhängig vom Modell nach übergeordneten Objekten (z.B. Gliederungspunkte, „Elternobjekte“) strukturiert abgebildet. In den sich so aufspannenden Zellen werden die Beziehungen vermerkt. Matrizen eignen sich, um alle Objekte und deren Beziehungen vollständig abzubilden und die vorhandenen Beziehungen zu analysieren (z.B. Clusterbildung).

Bei der oft sehr großen Anzahl von Objekten in technischen Produkten sind Matrizen allerdings wenig geeignet, um die initialen Beziehungen zu setzen. Bei der Entwicklung von Kraftfahrzeugen kann allein von mehreren Zehntausend Anforderungen ausgegangen werden [Vie09]. Selbst ein kleines Projekt kann schon über fünfzig Objekte beinhalten. Falls die Überprüfung pro Zelle durchschnittlich eine Minute beansprucht, wäre ein Mitarbeiter eine Woche mit dem Ausfüllen der Matrix beschäftigt. Die Wahrscheinlichkeit, dass bei dieser (stumpfsinnigen) Arbeit Fehler und Fehleinschätzungen begangen werden ist groß. Daher erscheint die Aussagekraft einer auf diese Art erstellten Matrix gering. Werden die Beziehungen im Modell immer gesetzt, wenn sie erkannt werden entsteht—sozusagen nebenbei—die Basis eines sinnvollen Beziehungssystems.

Zur Vervollständigung des Beziehungssystems wird die Matrix rechnerunterstützt ausgewertet. Es wird überprüft, ob die in der Matrix vorhandenen Basisbeziehungen auf weitere Beziehungen schließen lassen.

Weiterhin gibt es den Spezialfall der Intramodellbeziehungen. Hier stehen sich dieselben Objekte in Kopfspalte und -zeile gegenüber. In der Literatur ist eine solche Matrix für das Partialmodell „Anforderungen“ auch in der Darstellung als Dreieckmatrix (Beeinflussungsmatrix o.ä.) zu finden. In $n \times n$-Matrix-Darstellung wird die Hauptdiagonale nicht mit Werten belegt. Wird die Matrix als Konven-

tion von links nach rechts gelesen, so steht in den Zeilen welche Objekte das jeweilige Zeilenobjekt beeinflusst. In den Spalten steht durch welche Objekte das Spaltenobjekt beeinflusst wird.

Kapitel 4

Konzept zur Anforderungsmodellierung

In diesem Kapitel wird ein Konzept zur Anforderungsmodellierung entwickelt. Anforderungen sind Ausgangspunkt und wesentlicher Kern der Produktentwicklung. Ihre systematische Erfassung, Verarbeitung und Bereitstellung sind Schlüsselaktivitäten, die über Erfolg oder Misserfolg der Entwicklung entscheiden.

Ziel des entwickelten Modellierungskonzepts ist es, den interdisziplinären Produktentwicklungsprozess zu unterstützen und die in Abschnitt 2.2 dargestellten Schwierigkeiten bei der Arbeit in Kooperationsnetzwerken zu entschärfen. Außerdem soll es sich eignen eine gezielte, anforderungsgesteuerte Baukastenentwicklung zu vereinfachen.

Zur Erreichung dieser Ziele ist eine Anforderungsmodellierung notwendig. *Kläger* gibt in [Klä93, S. 114] die folgende Definition für diesen Begriff an:

> *Die Anforderungsmodellierung ist ein mehrstufiger, konstruktionsphasenübergreifender und dynamischer Prozess, dessen Zielsetzung und Inhalt ausgehend von einer „initialen" Konstruktionsaufgabe gekennzeichnet ist:*
>
> 1. *durch eine möglichst präzise, operationale Definition der angestrebten Real-Eigenschaften des zu entwickelnden Produktes unter Berücksichtigung aller relevanten Einflussgrößen;*
> 2. *durch eine zielgerichtete, adäquate Repräsentation der jeweils relevanten Produktanforderungen in allen Konkretisierungsstufen des Konstruktionsprozesses;*
> 3. *durch eine bewertungsgerechte Ableitung der als fest definierten Anforderungen zu Bewertungskriterien eines Kriteriensystems und*
> 4. *durch eine dynamische Ergänzung (Modifikation) von zusätzlichen, während des Entwicklungsvorgangs neu aufgetretenen Produktanforderungen.*

Humpert definiert das Ergebnis der Anforderungsmodellierung—das Anforderungsmodell—in [Hum95, S. 8] folgendermaßen:

> *Ein Anforderungsmodell ist ein Teil eines Produktmodells und bildet als Träger für Anforderungsinformationen alle geforderten charakteristischen Merkmale und Daten eines zukünftigen Produktes über den gesamten Produktlebenszyklus ab. Es beschreibt eine strukturelle Anordnung von Anforderungen einer ungeordneten Anforderungsmenge und bildet den Bezug zu den übrigen Partialmodellen des Produktmodells.*

Den Definitionen von *Kläger* und *Humpert* folgend wird ein Konzept erarbeitet, in dem das Anforderungsmodell einen Bezug zu den übrigen Partialmodellen des Produktmodells bildet. Dabei geht es von einer „initialen Aufgabe“ (Systemidee) aus und wird dynamisch an den aktuellen Stand der Entwicklung angepasst.

In Abb. 4.1 sind die in Abschnitt 2.1 identifizierten allgemeinen Bausteine einer interdisziplinären Produktentwicklung zusammengefasst dargestellt. Für eine zielgerichtete Entwicklung müssen die für jedem Baustein erarbeiteten Aktivitäten durchgeführt werden. Zur Unterstützung der Aktivitäten wurden Partialmodelle entwickelt, die in den folgenden Abschnitten detailliert beschrieben werden. Für einige Bausteine werden mehrere Partialmodelle verwendet. Mit dem Anforderungsmodell wird der Kern der Produktentwicklung, d.h. das Erfassen, Verarbeiten und Bereitstellen von Anforderungen abgedeckt. Für die Modellierung des Systemkontexts ist hier lediglich ein Partialmodell angegeben. Im konkreten Fall sind verschiedene projektspezifische Modelle—insbesondere in den späteren Entwicklungsphasen—notwendig. Die SysML bietet bereits umfangreiche Möglichkeiten und aktuelle Forschungsprojekte sind bestrebt hier weitere Verbesserungen zu erzielen (z.B. [La 06] [Joh08] [Alv09]).

Die Partialmodelle werden in der in Abb. 4.1 gezeigten Reihenfolge in Abschnitt 4.1 beschrieben. Im Sinne der Entwicklungskugel (vgl. Abschnitt 2.1) bildet diese Reihenfolge lediglich ein erstes Skizzieren des Modells ab. Die Ausarbeitung der einzelnen Partialmodelle erfolgt jeweils erst dann, wenn es im Projektverlauf relevant ist. Gleichzeitig werden die Partialmodelle immer nur so weit ausgearbeitet, wie es für den nächsten Schritt der Entwicklung notwendig erscheint. Die Reihenfolge, in der die Partialmodelle verwendet werden ist dabei nicht a priori festgelegt, sondern flexibel an die jeweiligen Bedürfnisse anzupassen.

Zur Unterstützung bestimmter Aktivitäten (z.B. Anforderungserhebung planen) werden Inter- und Intramodellbeziehungen untersucht (vgl. Abschnitt 3.4.2). Dazu werden insbesondere die in Abschnitt 4.2 beschriebenen Auswertematrizen herangezogen.

In Abschnitt 3.2 wurden bereits verschiedene allgemeine Partialmodelle vorgestellt. Als sinnvolle Beschreibungssprache wurde die SysML identifiziert und in Abschnitt 3.3 genauer beschrieben. In diesem Kapitel wird die Abbildung der hier wesentlichen Partialmodelle durch eine erweiterte SysML-Notation dargestellt, sowie Möglichkeiten der Auswertung und Weiterverarbeitung betrachtet.

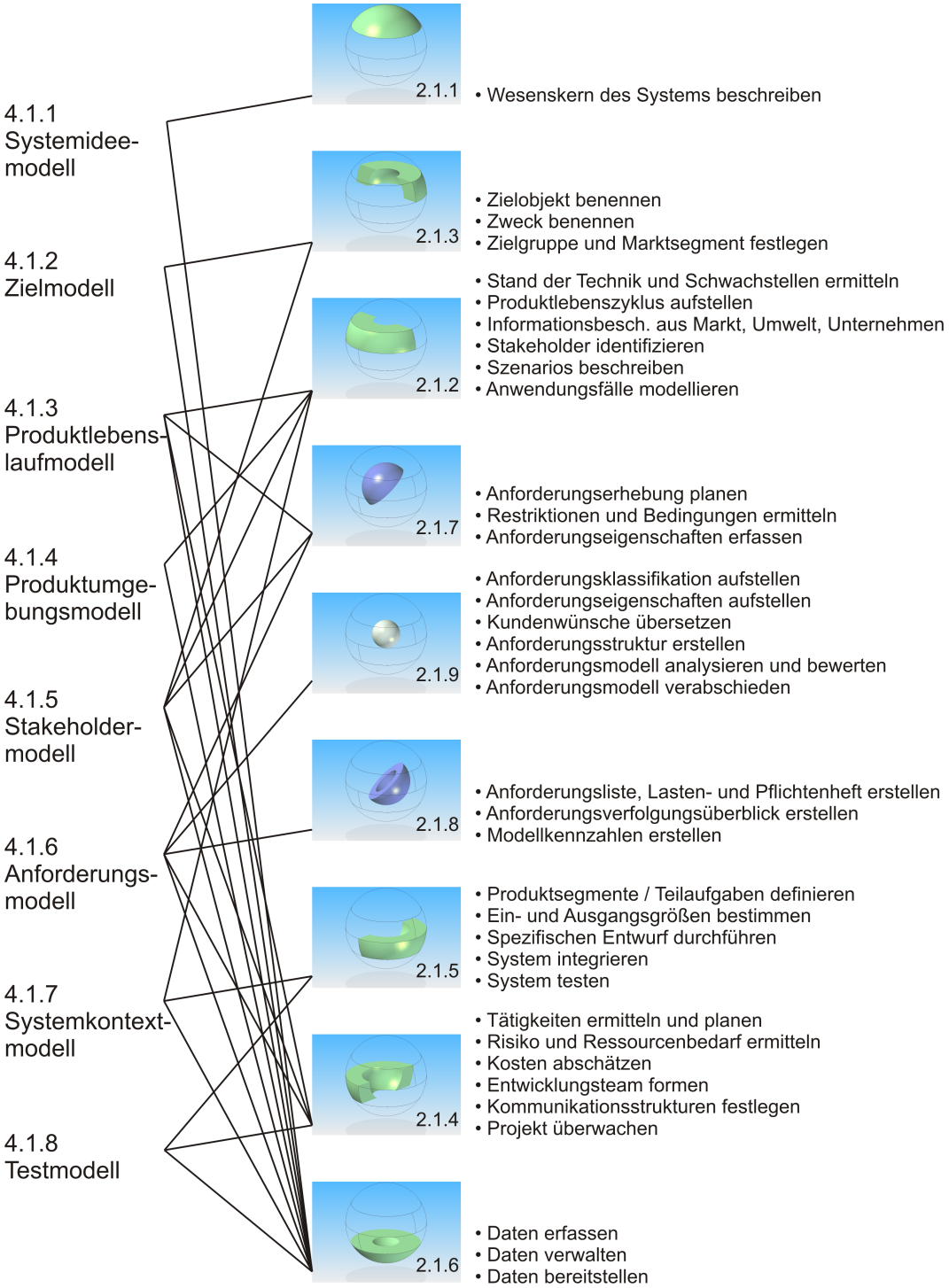


Abbildung 4.1: *Partialmodelle zur Produktmodellierung (links) und flexiblen Unterstützung der Aktivitäten im interdisziplinären Produktentwicklungsprozess (rechts).*

Abschließend wird eine Vorgehensweise für eine gezielte, interdisziplinäre und anforderungsgesteuerte Entwicklung von Baukastensystemen vorgeschlagen. Ein konkretes Anwendungsbeispiel folgt in Abschnitt 5 und verdeutlicht das hier beschriebene Konzept und die Vorgehensweise.

4.1 Verwendete Partialmodelle

Als Vorgehen im Sinne der beschriebenen Entwicklungskugel (vgl. Abb. 2.2) werden die einzelnen im folgenden beschriebenen Partialmodelle nicht nacheinander bearbeitet, sondern parallel sukzessive konkretisiert. Ein Partialmodell muss immer so konkret sein, wie es der nächste Schritt erfordert, d.h. es soll weder vorgreifen noch dürfen Lücken auftreten. In der Regel ist eine iterative Konkretisierung erforderlich. Der Projektfortschritt soll sich als „Wellenfront“ gleichmäßig ausbreiten.

Abb. 4.2 zeigt einen Screenshot des Package-Browsers für das Modell eines Parallelroboters in Artisan Studio Version 7. Das Modell beinhaltet die im Programm bereits vorgesehenen Profile `UML` und `SysML` und das für diese Arbeit neu entwickelte Profil `SFB562`, welches die neuen *tag definitions* und *stereotypes* beinhaltet. Die folgenden Ordner enthalten jeweils die Objekte der entsprechenden Partialmodelle. Das Stakeholdermodell, der Produktlebenslauf und die Produktumgebung sind in dem Ordner *Situation* zusammengefasst. Die testrelevanten Objekte und die *change notes* des *Changemanagements* sind im Ordner *Projektüberwachung* enthalten. Die Packagestruktur entspricht somit der in Abb. 4.1

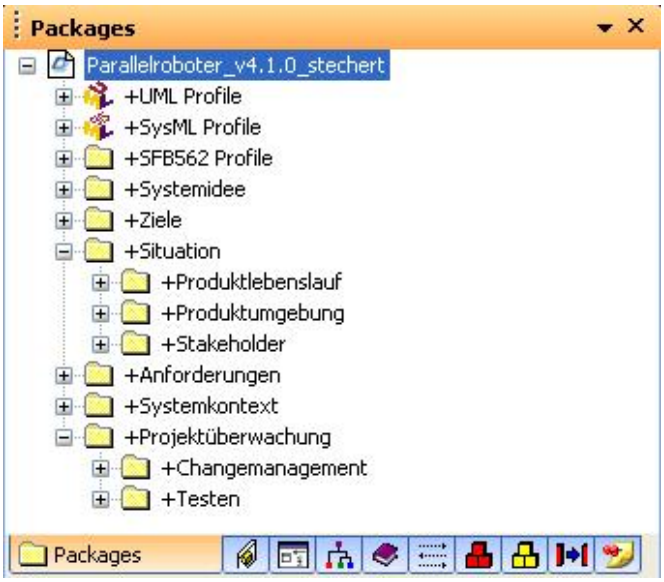


Abbildung 4.2: *Screenshot des Package-Browsers in Artisan Studio für das Modell eines Parallelroboters.*

dargestellten Struktur.

In den folgenden Erläuterungen der einzelnen Partialmodelle werden die Erweiterungen und Veränderungen zur als bekannt vorausgesetzten SysML und dem Entwicklungswerkzeug Artisan Studio thematisiert. Insbesondere wird der Nutzen der Modelle für die Unterstützung der für die Produktentwicklung notwendigen Aktivitäten herausgestellt.

4.1.1 Systemideemodell

Die Systemidee (vgl. Abschnitt 2.1.1) beschreibt auf sehr abstrakter Ebene den Wesenskern dessen, was entwickelt werden soll. Dabei ist die Systemidee noch vollkommen unabhängig von der späteren Umsetzung.

Die Systemidee kann sich aus mehreren Objekten zusammensetzen. Diese werden in Form der SysML-Klasse `<<requirement>>` erzeugt und mit dem neuen Stereotyp `<<goal>>` versehen. Zur schnelleren Unterscheidbarkeit wird dem Objektnamen ein `g_` vorangestellt. Der Stereotyp `<<goal>>` enthält die folgenden *Tag Definitions*:

- `branch`: Enthält die Branche(n), der diese Systemidee zuzuordnen ist, z.B. Feinwerktechnik.
- `source`: Enthält die Quelle bzw. den Ursprung der Idee, z.B. Betreiber oder Hersteller.
- `author`: Der Autor des Objekts, d.h. die natürliche Person oder die Arbeitsgruppe, die das Objekt zum ersten Mal formuliert.
- `responsable`: Der Verantwortliche für das Objekt ist die natürliche Person, die für die korrekte Formulierung des Objekts zuständig und für dessen Erfüllung verantwortlich ist.
- `state`: Der Objektstatus unterscheidet zwischen:
  - `created`: Das Objekt wurde erstellt, ist aber noch nicht formal vollständig beschrieben, d.h. nicht alle Merkmale sind ausgefüllt und nicht alle notwendigen Beziehungen gesetzt.
  - `signed`: Das Objekt ist formal vollständig beschrieben.
  - `revise`: Das als `signed` beschriebene Objekt wurde überprüft und nicht freigegeben. Es muss von einer verantwortlichen Person überarbeitet werden.
  - `proofed`: Das als `signed` beschriebene Objekt wurde überprüft und freigegeben.

– `refused`: Das Objekt wird zurückgestellt. Dieser Status ist gleichbedeutend mit einer Löschung, allerdings steht das Objekt weiterhin zur Dokumentation der Entscheidungsprozesse zur Verfügung. So wird vermieden, dass nicht versehentlich das gleiche Objekt zum wiederholten Male erzeugt und nach nochmaligem Durchlaufen des gleichen Entscheidungsprozesses (vielleicht durch andere Personen) erneut gelöscht wird. Entwicklungsprozesse verlaufen effizienter, da unnötige Iterationen vermieden werden.

In Abb. 4.3 wird in dem Anforderungsdiagramm `[Systemidee] Flexibilität` das Objekt `g_hohe Flexibilität` der Systemidee für ein Handhabungs- und Montagesystem visualisiert. Auf dieses Objekt werden die beiden Stereotypen `<<requirement>>` und `<<goal>>` angewendet. Zur besseren Unterscheidung von Systemidee, Zielen und Anforderungen ist dem *Stereotyp* `<<goal>>` die Hintergrundfarbe grün zugeordnet worden. Im unteren Bereich der Darstellung sind die oben angegebenen *Tag Definitions* als *Compartments* gezeigt. Dabei sind die Merkmale als Überschrift in fetter Schrift und die Merkmalsausprägungen darunter eingetragen.

Beziehungen zum Stakeholdermodell sind durch die Merkmale Autor und Verantwortlicher abgebildet, denn diese sind als natürliche Personen im Stakeholdermodell erfasst. Die Merkmalsausprägung enthält hier eine Referenz zu diesem weiteren Partialmodell (Äquivalenzbeziehung, Abschnitt 3.4.2). Die Quelle ist ebenfalls eine der Rollen des Stakeholdermodells, allerdings in diesem Fall keine natürliche Person, sondern eine übergeordnete abstrakte Rolle (vgl. Abschnitt 4.1.5). Für das Merkmal Branche werden Referenzen zu den entsprechen-

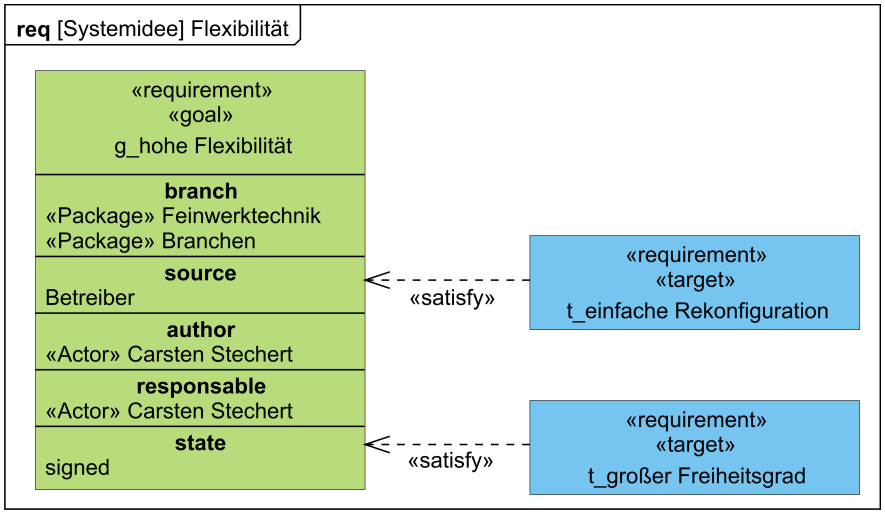


Abbildung 4.3: *„Flexibilität“ als visualisiertes Teilobjekt der Systemidee in einem Anforderungsdiagramm, sowie zwei ausgewählte, dieses Objekt erfüllende, Ziele.*

den *Packages* des Produktlebenslaufmodells erzeugt (vgl. Abschnitt 4.1.3).

Weiterhin bestehen `satisfy`-Beziehungen zu Zielen des Zielmodells. Die Ziele tragen dazu bei, bestimmte Objekte der Systemidee zu erfüllen (Erfüllungsbeziehung, Abschnitt 3.4.2). In Abb. 4.3 sind zwei ausgewählte Ziele dargestellt, die zur Erfüllung einer hohen Flexibilität beitragen: `t_einfache Rekonfiguration` und `t_großer Freiheitsgrad` (mit `t_` als Kennzeichnung eines Zieles). Durch Rekonfiguration kann ein Robotersystem flexibel an veränderte Randbedingungen angepasst werden. Ein großer Freiheitsgrad erlaubt es, die Handhabungsobjekte auf sehr unterschiedliche Weise (Richtungen, Orientierungen) zu manipulieren.

4.1.2 Zielmodell

Ziele beschreiben abstrakt die Zielrichtung der Entwicklung (s. Abschnitt 2.1.3). Dabei sind Ziele bereits an bestimmte Strategien (z.B. Einsatz eines Baukastensystems) gekoppelt, aber von einer konkreteren Systemlösung unabhängig. Es werden Zielobjekt und Zweck benannt und durch die modellierten Ziele genauer spezifiziert. Die hierfür zugrunde gelegte Zielgruppe und das Marktsegment werden im Stakeholdermodell (s. Abschnitt 4.1.5) dokumentiert und entsprechend referenziert.

Ziele sollen zu jeder Zeit allen Projektmitarbeitern bekannt sein und alle Arbeiten sollen zur Erfüllung dieser Ziele beitragen (vgl. Abschnitt 2.2.3). Bildlich gesprochen sind es die Stränge an denen das Entwicklungsteam gemeinsam zieht.

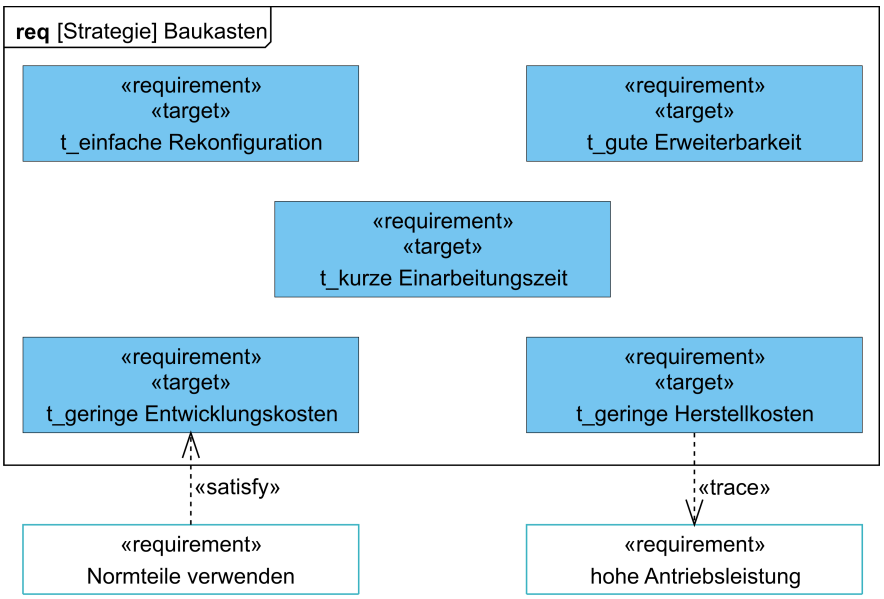


Abbildung 4.4: *Visualisierte Ziele der Strategie „Baukastenentwicklung“ in einem Anforderungsdiagramm, sowie zwei beispielhafte, die Ziele erfüllende (satisfy) bzw. konfliktäre (trace), Anforderungen.*

Sie bilden das Bindeglied zwischen der Systemidee und den Anforderungen.

Die Ziele werden als `<<requirement>>` erzeugt und mit dem neuen Stereotyp `<<target>>` versehen. Zur schnelleren Unterscheidbarkeit wird dem Namen ein `t_` vorangestellt. Der Stereotyp `<<target>>` enthält die bereits in Abschnitt 4.1.1 vorgestellten *Tag Definitions* `branch`, `source`, `author`, `responsable` und `state`. Die Beziehungen zum Stakeholdermodell und Produktlebenslaufmodell gleichen denen des Systemideemodells.

Abb. 4.4 zeigt in dem Anforderungsdiagramm `[Strategie] Baukasten` fünf Ziele als Objekte, die der Strategie „Baukastenentwicklung“ zugeordnet werden (vgl. Stoßrichtungen nach *Renner* Abb. 2.14 in Abschnitt 2.3).

Wie bereits in Abb. 4.3 zu sehen war, bestehen `satisfy`-Beziehungen zu den Objekten des Systemideemodells. Die Ziele tragen dazu bei, bestimmte Objekte der Systemidee zu erfüllen. Zudem können Anforderungen die Ziele erfüllen. In Abb. 4.4 erfüllt die Anforderung `Normteile verwenden` das Ziel `t_geringe Entwicklungskosten`. `trace`-Beziehungen zu Anforderungen geben hingegen im Allgemeinen eine negative Beziehung an. Falls ein Ziel über eine `trace`-Beziehungen auf eine Anforderung zeigt, so wird die Erfüllung der Anforderung negativ auf die Zielerfüllung einwirken. Beispielsweise führt die Forderung nach hoher Antriebsleistung im Allgemeinen zu teuren Antrieben. Diese werden als Kaufteile in den Herstellkosten berücksichtigt und wirken dem Ziel der geringen Herstellkosten entgegen.

4.1.3 Produktlebenslaufmodell

Der Produktlebenslauf beschreibt alle Phasen, die während des Produktlebens durchlaufen werden. Das Produktlebenslaufmodell unterstützt die Aktivität des Aufstellens des Produktlebenszyklus' und der Beschreibung von (derzeitigen und zukünftigen) Szenarien (vgl. Abschnitt 2.1.2). Zusammen mit dem Stakeholder-

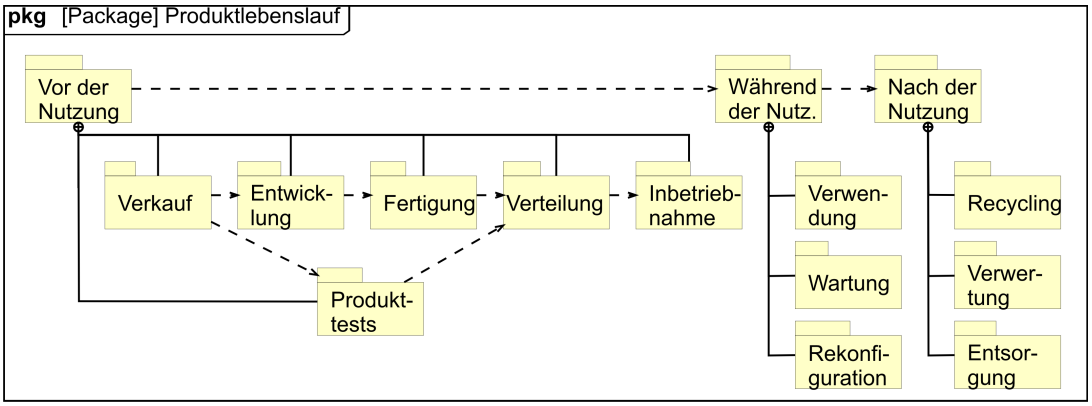


Abbildung 4.5: *Phasen des Produktlebenslaufs als Packages in zeitlicher und logischer Anordnung.*

modell bildet es die Grundlage für die Planung der Anforderungserhebung. Werden in diesem Partialmodell ebenfalls die Anwendungsfälle der Entwicklung erfasst, so ist es für die Projektplanung nutzbar. Die einzelnen notwendigen Tätigkeiten werden ermittelt bzw. geplant. Zusammen mit dem Stakeholdermodell wird der Ressourcenbedarf ermittelt. Außerdem lassen sich Kommunikationsstrukturen festlegen.

Die Lebenslaufphasen sind als eigene *Packages* (vgl. „Verzeichnis“ in der ordnenden Datenstruktur der digitalen Datenverarbeitung) angeordnet und stehen in zeitlicher oder logischer Beziehung zueinander. In Abb. 4.5 sind diese Packages dargestellt. Die gestrichelten Pfeile geben eine zeitliche Abhängigkeit an. Beispielsweise ist die Phase *Verkauf* zeitlich im Allgemeinen der Phase *Produkttests* vorgelagert, da zunächst Abnahmekriterien definiert und ggf. vertraglich festgehalten werden müssen bevor sie getestet werden. Die Phase der *Verteilung* beginnt ihrerseits erst, wenn *Fertigung* und *Produkttests* abgeschlossen sind. Gleichzeitig sind die genannten Packages logisch der übergeordneten Phase *Vor der Nutzung* zugeordnet.

In den einzelnen Phasen befinden sich SysML-Anwendungsfälle (*Use Cases*), die die Tätigkeiten innerhalb der Phase genauer beschreiben. Anwendungsfälle besitzen eine Vor- und eine Nachbedingung, sowie eine Beschreibung der Durchführung, d.h. dessen was zwischen Vor- und Nachbedingung passiert.

Anwendungsfälle können wie in Abb. 4.6 gezeigt innerhalb des Produktle-

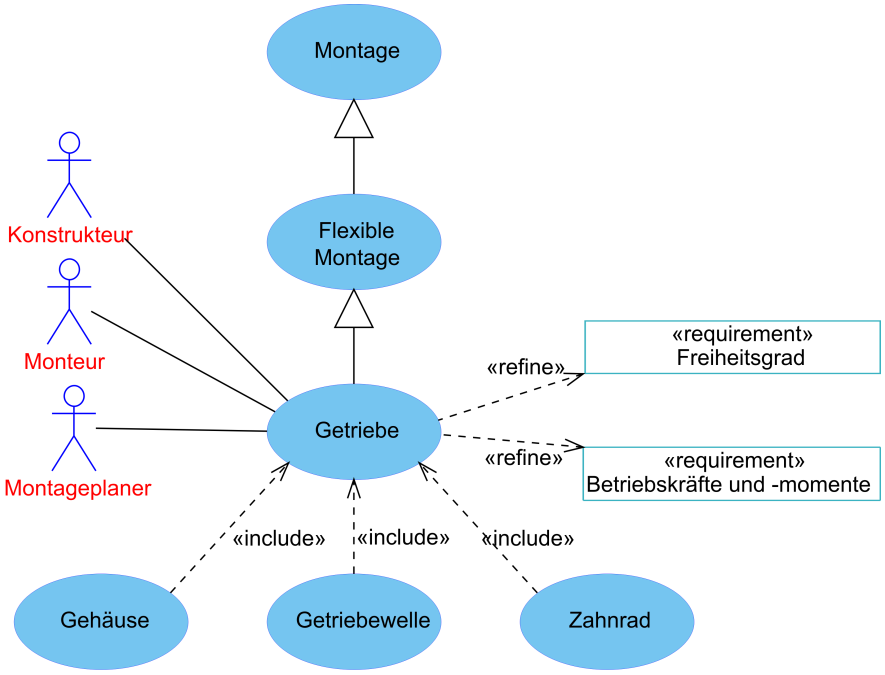


Abbildung 4.6: *Anwendungsfall der Montage eines feinwerktechnischen Getriebes und damit zusammenhängende Objekte.*

benslaufmodells miteinander in Beziehung stehen. Die in SysML beschriebenen `extend`- bzw. `include`-Beziehungen geben an, dass ein Anwendungsfall durch einen anderen erweitert wird bzw. das ein Anwendungsfall in einem weiteren enthalten ist (vgl. Dekomposition, Abschnitt 3.4.2). So beinhaltet die Montage eines Getriebes unter Anderem das Fügen von Wellen und Zahnrädern und das Einlegen der vormontierten Getriebewellen in das Gehäuse. Weiterhin wird durch Generalisierung eine Beziehung zu einem übergeordneten Anwendungsfall beschrieben. In Abb. 4.6 ist dies die flexible Montage, die wiederum Spezialisierung der Montage ist. Die Zuordnung zu bestimmten Branchen (hier z.B. Feinwerktechnik) erfolgt über die Einordnung der Anwendungsfälle in entsprechend benannte Packages. Diese sind als Unterordner von *Während der Nutzung*, *Verwendung* (vgl. Abb. 4.5) und schließlich *Anwendung* angelegt.

Außerdem gibt es Beziehungen zum Stakeholdermodell (s. Abschnitt 4.1.5). In Abb. 4.6 sind drei Rollen (z.B. Konstrukteur) dargestellt und mit dem zentralen Anwendungsfall in Beziehung gesetzt. Im konkreten Fall können hier ebenfalls natürliche Personen des Stakeholdermodells angegeben werden. Weiterhin sind `refine`-Beziehungen (Herkunftsbeziehungen, vgl. Abschnitt 3.4.2) zu Anforderungen des Anforderungsmodells (s. Abschnitt 4.1.6) gezeigt. Ein bestimmter Anwendungsfall erlaubt es, eine Anforderung detaillierter zu fassen. Hier definiert die konkrete Montageaufgabe die notwendigen Freiheitsgrade des Montagesystems und die bereitzustellenden Betriebskräfte und -momente zur Aufgabendurchführung.

4.1.4 Produktumgebungsmodell

Die Produktumgebung umfasst die angrenzenden Systeme, die in unterschiedlicher Weise Einfluss auf das zu entwickelnde Produkt haben (vgl. Abschnitt 2.1.2). Je nach Relevanz werden im Produktumgebungsmodell die Umgebung bzw. Nachbarsysteme mehr oder weniger detailliert modelliert. Das Modell dient im Wesentlichen der Informationsbeschaffung aus Umwelt und Unternehmen.

In einem ersten Schritt werden relevante Objekte der Produktumgebung als `<<block>>` der SysML dargestellt. Durch in der SysML vorgesehene Assoziationen werden die Blöcke miteinander verbunden. Einen besonderen Stellenwert erhalten hier z.B. Nachbarsysteme, von denen im konkreten Fall Störeffekte auf das zu entwickelnde System ausgehen. Abb. 4.7 zeigt das *Block Definition Diagramm* `[Produktumgebung] Anlagenebene` als Überblickdarstellung der Produktumgebung eines Robotersystems. Je nach konkreter Umgebungssituation werden die Objekte stärker detailliert. So kann ein Fördermittel z.B. ein Stetigförderer mit Doppelgurtbahn oder ein Unstetigförderer als fahrerloses Transportsystem sein. Diese Möglichkeiten werden als Objekte im Modell dokumentiert und mit Spezialisierungsbeziehungen zu dem allgemeineren Objekt versehen. Durch die SysML-Beziehungen `Aggregation` und `Composition` werden—falls notwendig—die Umgebungsobjekte in Komponenten zerlegt. Außerdem werden Blöcke bei Bedarf mit

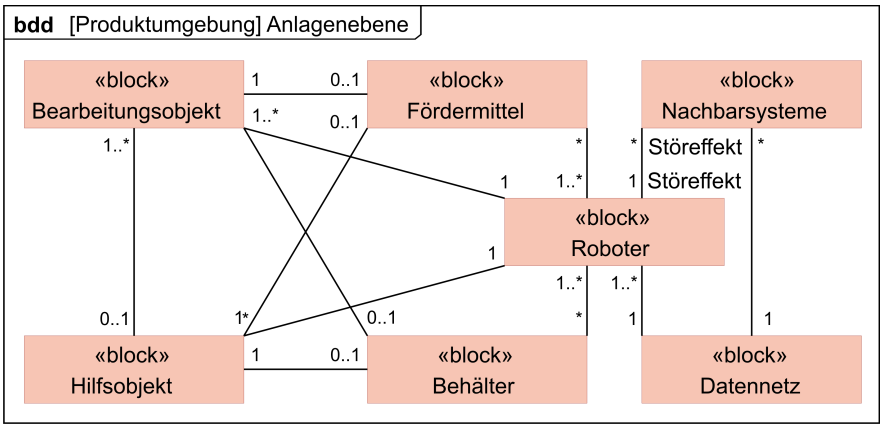


Abbildung 4.7: *Abstrakte Darstellung eines Robotersystems innerhalb der Produktumgebung in einem „Block Definition Diagram“ (bdd).*

Flow Ports ausgerüstet, die Flüsse bzw. Schnittstellen zwischen Blöcken genauer darstellen und mit Dimensionen und ggf. physikalischen Maßeinheiten belegen.

Die SysML bietet hier viele Möglichkeiten, insbesondere die regelungstechnischen Aspekte sehr detailliert zu modellieren. Für das Produktumgebungsmodell muss stets im Einzelfall entschieden werden, welcher Nutzen einem stärkeren Modellierungsaufwand gegenübersteht. Im Allgemeinen reicht es aus, die in Beziehung stehenden Objekte und ihre relevanten Parameter zu identifizieren. Die Modellierung der Schnittstellen erfolgt in Form von *Stubs*. Diese „Stummel“ stellen lediglich Eingangswerte bereit und nehmen Ausgangswerte an, ohne weitere Funktionalitäten zu besitzen.

Einen besonderen Stellenwert nimmt die Produktumgebung beim Testen ein. Das ursprünglich zur Aufgabenklärung und Analyse verwendete Modell bildet dann die Basis für die Ausgestaltung der Testumgebung (z.B. Software-in-the-Loop, vgl. auch Abschnitt 4.1.8).

4.1.5 Stakeholdermodell

Das Stakeholdermodell hilft die involvierten Stakeholder zu identifizieren und zu strukturieren (vgl. Abschnitt 2.1.2). Stakeholder sind die Personen bzw. Personengruppen und Rollen, die in irgendeiner Weise (aktiv oder passiv) Einfluss auf das Produkt bzw. die Produktentwicklung nehmen. In diesem Partialmodell wird die Zielgruppe und damit das Marktsegment genauer festgelegt. Wie bereits dargelegt, bildet es zusammen mit dem Produktlebenslaufmodell die Grundlage für eine systematische Planung der Anforderungserhebung. Im Bereich der Projektplanung wird es verwendet, um den Ressourcenbedarf zu ermitteln, das Entwicklungsteam zu formen und Kommunikationsstrukturen festzulegen (vgl. Abschnitt 2.1.4, Abschnitt 2.2.1, Abschnitt 2.2.5).

Es wird der in der SysML vorhandene *Type* `Actor` als Basis verwendet. Dieser

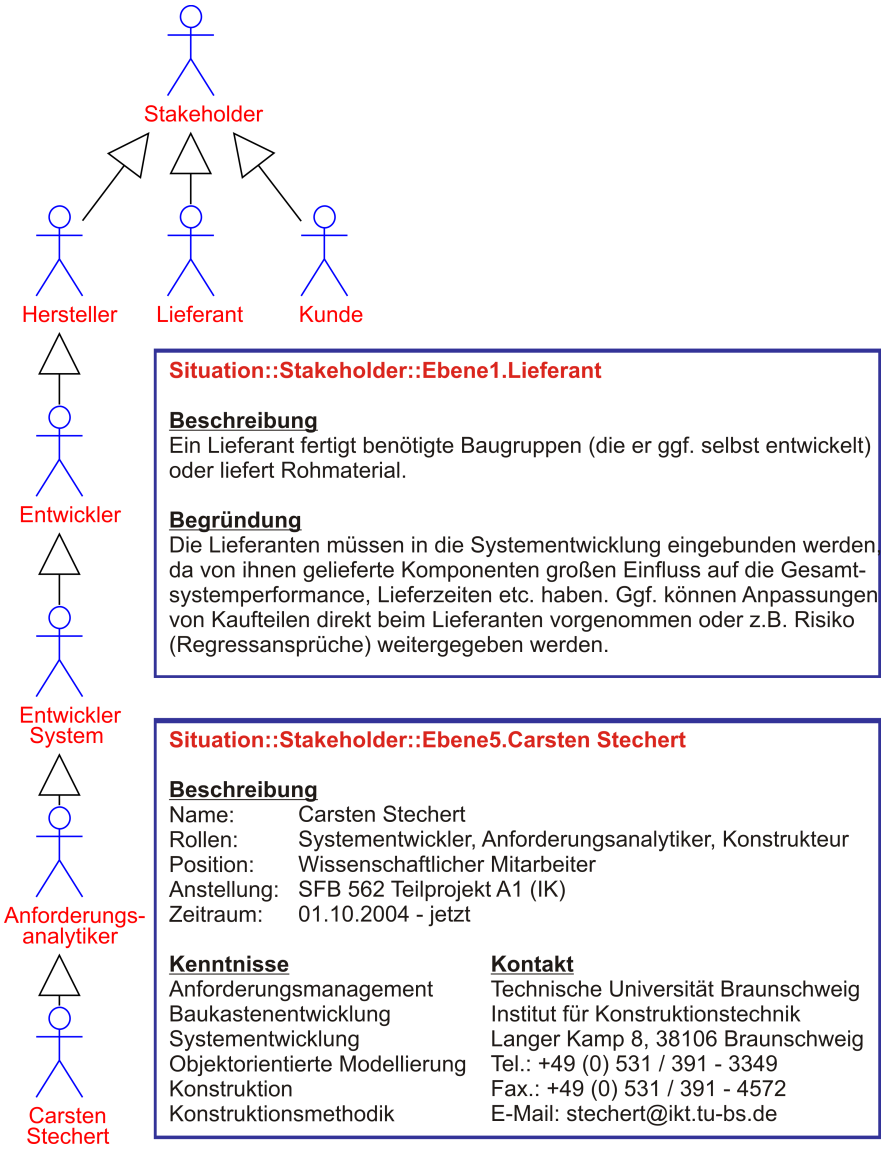


Abbildung 4.8: *Ausschnitt der hierarchischen Stakeholderstruktur und Beschreibung von zwei Objekten.*

erhält als neue Information eine Beschreibung seiner Aufgabe und eine Begründung für die Beteiligung dieses Stakeholders an der Produktentwicklung. Stakeholder können auf unterschiedlichen Ebenen betrachtet werden. In dem entwickelten Modell wird eine Einteilung in fünf Ebenen vorgenommen und die Stakeholderobjekte entsprechenden Packages zugeordnet. Die Objekte einer Ebene sind durch Generalisierungen mit einem Objekt der jeweils höheren Ebene verknüpft. Auf oberster Ebene—der Ebene Null—steht die Rolle Stakeholder als höchst mögliche Generalisierung. Darunter—auf Ebene 1—stehen abstrakte Rollen wie Hersteller, Lieferant oder Kunde. Diese Rollen werden sukzessive spezialisiert bis

schließlich auf der fünften Ebene natürliche Personen stehen. Diese enthalten in ihrer Beschreibung sämtliche notwendige Kontaktdaten, mit Position und Zeitraum der Projektbeteiligung. Außerdem werden die Rollen eingetragen, die von der Person eingenommen werden. Weiterhin werden die speziellen Kenntnisse der Person dokumentiert, die im Rahmen des Projekts von Bedeutung sind. Damit ist dieser Teilaspekt ein wichtiges Hilfsmittel für die Projektplanung (vgl. Abschnitt 2.1.4).

In Abb. 4.8 ist eine ausgewählte Kette von Generalisierungen dargestellt, ausgehend von einer natürlichen Person auf Ebene fünf, über die Rollen Anforderungsanalytiker und Entwickler bis hin zur Rolle Hersteller auf Ebene eins und schließlich Stakeholder auf Ebene Null. Im unteren rechten Bereich sind die hinterlegten Informationen für die natürliche Person gezeigt. Es ist eine allgemeine Beschreibung angegeben, die den Namen, die im Projekt eingenommenen Rollen, die aktuelle Position, die Anstellungsdetails ggf. mit Angabe des Linienvorgesetzten und den Anstellungszeitraums enthält. Letzteres lässt für eine sinnvolle Aufgabenverteilung auf die Erfahrung des Mitarbeiters schließen. Weiterhin werden die projektrelevanten Kenntnisse (z.B. im Anforderungsmanagement oder der Baukastenentwicklung) und die genauen Kontaktdaten der Person dokumentiert.

Oberhalb der vorgestellten Beschreibung befinden sich die Informationen für die Rolle „Lieferant“ mit Rollenbeschreibung und Begründung für die Einbeziehung der Rolle in den Entwicklungsprozess. Auf der hier gezeigten Ebene 1 fallen die Beschreibungen noch sehr allgemein aus. Sie haben den Charakter einer Definition, so dass innerhalb des Projekts stets ein einheitlicher Wortgebrauch stattfindet. Gleiches gilt für die Begründung.

Wie bereits in den vorherigen Abschnitten aufgezeigt, werden Stakeholder in weiteren Partialmodellen genutzt. In den Modellen Systemidee, Ziele, Anforderungen und Test erscheinen sie als Autoren und Verantwortliche oder Informationsquellen. Hier meist als natürliche Personen der Ebene fünf angegeben. Außerdem erscheinen sie im Produktlebenslauf- und Testmodell als durchführende Akteure von Anwendungs- und Testfällen.

Zu Beginn einer Entwicklung sind oftmals nicht sämtliche Stakeholder im Sinne von natürlichen Personen bekannt. Insbesondere in den Unternehmen des Kunden, Zulieferers, Serviceanbieters oder der Kunden des Kunden sind—abgesehen von Vertriebsmitarbeitern—kaum natürliche Personen als Ansprechpartner zu identifizieren. Die verschiedenen Rollen sind als Muster allerdings in nahezu jedem Unternehmen zu finden. Die Aufteilung von natürlichen Personen auf die Rollen ist dann unternehmensspezifisch.

4.1.6 Anforderungsmodell

Das Anforderungsmodell steht—wie in Abb. 3.3 gezeigt—mit sämtlichen Partialmodellen in Beziehung. Außerdem unterstützt es die drei Bausteine des Anforderungsmanagements (2.1.7 – 2.1.9, vgl. Abb. 4.1) und die Projektüberwachung

(vgl. Abschnitt 4.2.8). Es werden alle in Interviews oder mit Hilfe von Checklisten ermittelten Anforderungen, Bedingungen und Restriktionen erfasst und—soweit möglich—die jeweiligen Anforderungseigenschaften dokumentiert. Anschließend werden die erfassten Anforderungen—falls notwendig—in technische Anforderungen übersetzt und die Anforderungseigenschaften vervollständigt. Des Weiteren werden die Anforderungen durch eine Klassifikation strukturiert (vgl. Abschnitt 3.4.1). Die weitere Strukturierung wird durch Beziehungen zwischen den Anforderungen (z.B. Dekomposition) und zu Objekten anderer Partialmodelle abgebildet (z.B. Verfeinerung durch Anwendungsfälle). Dieses strukturierte Modell ermöglicht es systematisch nach Redundanzen, Konflikten und unnötigen oder unpräzisen Anforderungen zu suchen (vgl. Abschnitt 4.2). Das Modell wird analysiert und ggf. umstrukturiert und bereinigt. Sobald das Anforderungsmodell verabschiedet und somit für weitere Entwicklungsschritte freigegeben wurde, bildet es die Basis für sichtenspezifische Anforderungslisten. Diese können wiederum genutzt werden, um Lasten- bzw. Pflichtenhefte auszuarbeiten. Für die Projektüberwachung werden aus dem Modell in festgelegten Abständen Kennzahlen ermittelt (vgl. Abschnitt 3.4.3, Abschnitt 4.2.8), sowie die Änderungen am Modell kontinuierlich erfasst und ggf. als Anforderungsverfolgungsüberblick bereitgestellt.

Die Basis des Anforderungsmodells bilden durch den SysML-Stereotyp `<<requirement>>` definierte Objekte. Dies ermöglicht es, bereits definierte *Tag Definitions* und Funktionalitäten zu nutzen. Die darüber hinausgehenden Attribute sind in dem neuen Stereotyp `<<ReqAttributes>>` zusammengefasst und werden im Folgenden kurz beschrieben. Durch die in der SysML vorgesehenen und die unten aufgeführten, zusätzlichen Attribute, werden die Anforderungen formal vollständig im Sinne der Aufstellung im Anhang Abschnitt A.2 erfasst bzw. bereitgestellt.

- `values and facts`: Enthält Werte und Daten zur genaueren Beschreibung der Anforderung. Im Idealfall wird dadurch die Anforderung quantitativ fassbar beschrieben, z.B. $l = 250\,\mathrm{mm}$.

- `annotations`: Enthält ggf. Anmerkungen als Hinweise für die weitere Entwicklung oder andere Entwickler.

- `priority`: Beschreibt die Priorität der Anforderung nach Fest-, Mindest- oder Wunschforderung.

- `kind`: Beschreibt die Art einer Anforderung hinsichtlich ihrer Klassifikation, z.B. nach Tabelle 3.1 und Tabelle 3.2.

- `quantifiable`: Anforderungen werden insbesondere in frühen Entwicklungsphasen häufig nur qualitativ angegeben. Im weiteren Verlauf sind oft auch quantitative Aussagen möglich. Daher wird hier (wie bei den Beziehungen,

vgl. Abschnitt 3.4.2) angegeben, ob die Anforderung qualitativ oder quantitativ vorliegt bzw. im späteren Verlauf quantifizierbar ist.

- `domain`: Beschreibt welche unterschiedlichen Domänen (Fachbereiche) diese Anforderung beeinflussen, z.B. Maschinenbau.

- `detailing`: Die Detaillierung beschreibt, welchem Detaillierungsgrad die Anforderung zuzuordnen ist. Dabei werden drei Stufen unterschieden und durch `hoch`, `mittel` und `gering` charakterisiert. Eine stärkere Aufspaltung wäre zwar möglich, erhöht jedoch die Komplexität ohne den Nutzen wesentlich zu steigern. Anforderungen eines geringen Detailierungsgrades eignen sich für eine qualitative, allgemeine Betrachtung auf Konzeptebene (z.B. „*einfache Fertigungsverfahren verwenden*“). Je weiter der Entwicklungsprozess fortschreitet, desto detailliertere Anforderungen sind notwendig (z.B. „*Als Blechbiegeteil ausführen*“). Je detaillierter die Anforderungen sind, desto stärker projektspezifisch sind sie. Anforderungen geringer Detaillierung können für ähnliche Entwicklungsaufgaben wiederverwendet werden.

- `certainty`: Gibt Auskunft darüber, ob die Aussagen (Werte/ Daten) der Anforderung als `sicher` oder `unsicher` eingeschätzt werden. Bei einer sicheren Anforderung wird davon ausgegangen, dass im weiteren Verlauf der Entwicklung keine Änderungen daran vorgenommen werden. Bei unsicheren Anforderungen werden im weiteren Verlauf die Werte wahrscheinlich angepasst oder die gesamte Anforderung umformuliert oder gelöscht.

- `branch`: Beschreibt die Branche, die wesentlichen Einfluss auf die Anforderung ausübt. Beispielsweise werden bei Produkten für unterschiedliche Branchen (z.B. Verpackungsmaschinen) die Hygiene betreffende Anforderungen häufig von der Lebensmittelindustrie vorgegeben, während chemische Beständigkeit der verwendeten Materialien häufig von der chemischen Industrie vorgegeben wird.

Außerdem enthält der Stereotyp `<<ReqAttributes>>` die bereits in Abschnitt 4.1.1 vorgestellten *Tag Definitions* `source`, `author`, `responsable` und `state`.

Die Anforderungen werden im Package Anforderungen (vgl. Abb. 4.2) in Unterordner strukturiert abgelegt. Durch die Unterordner wird eine Gliederung nach Merkmalen (z.B. nach Tabelle 3.1 und Tabelle 3.2) abgebildet.

In Abb. 4.9 ist ein Anforderungsdiagramm dargestellt, dass einige ausgewählte Objekte des Anforderungsmodells zeigt. Die Anforderungsobjekte sind mit den SysML-Stereotypen `<<requirement>>` gekennzeichnet. In der rechten Hälfte mittig ist die Anforderung `max. Nennlast` abgebildet. Nicht dargestellt sind die Ausprägungen der Merkmale im Stereotyp `<<ReqAttributes>>` (z.B. `priority: Festanforderung`, `quantifiable: quantitative`).

Gezeigt ist die bereits in Abschnitt 4.1.2 beschriebene Erfüllungsbeziehung

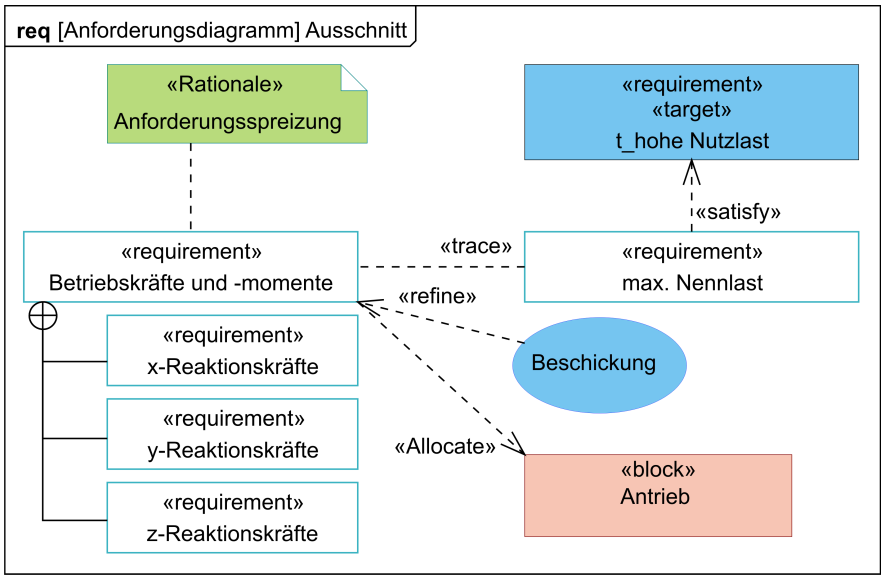


Abbildung 4.9: *Ausschnitt eines Anforderungsdiagramms mit verschiedenen in Beziehung stehenden Objekten sowie der Dekomposition einer Anforderung.*

zu den Zielen (hier `t_hohe Nutzlast`). Als weitere Beziehung ist eine `trace` zur Anforderung `Betriebskräfte und -momente` gesetzt. Nach *Rupp* müssen Beziehungen (*traces*) zwischen Anforderungen einige Voraussetzungen erfüllen [Rup07, S. 411f]. Sie sollen eindeutig identifizierbar sein und die Struktur und der Inhalt müssen definiert sein. Außerdem müssen die Ziele der Verfolgbarkeit (z.B. prozess-, physikbeschreibend) klar sein und der Aufwand gegenüber dem Nutzen abgewogen werden. Das Arbeiten mit Beziehungen ist nur dann sinnvoll, falls Beziehungen sofort in dem Moment während der Modellierung gesetzt werden, in dem sie dem Entwickler bewusst werden, und nicht etwa nachträglich durch das Ausfüllen von großen Matrizen. Außerdem müssen sie ständig gepflegt werden.

Um die oben angegebenen Voraussetzungen zu erfüllen, ist die `trace`-Beziehung durch die in Abschnitt 3.4.2 und Abschnitt 4.2.6 angegebenen Merkmale genauer definiert. Diese sind als *tag definition* dem Stereotyp `<<trace>>` hinzugefügt. Beispielsweise handelt es sich um eine wechselseitig positive Unterstützung, die die Anforderungen direkt in Beziehung setzt. D.h. wird eine Struktur für höhere Nennlasten ausgelegt, wird sie ebenfalls höhere Betriebskräfte bereitstellen können. Falls eine Struktur für höhere Betriebskräfte ausgelegt wird, so wird sie ebenfalls größere Nennlasten tragen können. Der genaue Einfluss ist allerdings im Einzelfall zu beurteilen, da z.B. Dynamikforderungen und die damit verbundenen Trägheitskräfte ebenfalls einen Einfluss haben.

Falls zwischen quantitativ beschreibbaren Anforderungen eine quantitativ beschreibbare Beziehung identifiziert wurde, wird diese durch ein `<<constraint>>` detailliert abgebildet. In dem durch die SysML bereitgestellten Stereotyp werden die notwendigen mathematischen Gleichungen, und Parameter angegeben. Die

Parameter werden durch `trace`-Beziehungen mit den entsprechenden Anforderungen verbunden (s. Abb. 4.10).

Die Anforderung `Betriebskräfte und -momente` ist mit einem Kommentar (`rationale`: *wörtlich* Begündung) versehen. Es weist darauf hin, dass für diese Anforderung eine Anforderungsspreizung (s. Abschnitt 2.3.1) vorliegt, d.h. bei unterschiedlichen Anwendungen nimmt diese Anforderung unterschiedliche Werte an. Beispielsweise werden bei Montageaufgaben hohe Betriebskräfte zum Fügen von z.B. Schnappverbindungen benötigt. Wird der gleiche Roboter für eine *pick and place* Aufgabe verwendet, sind die geforderten Betriebskräfte deutlich kleiner.

Die Ursachen der Anforderungsspreizung werden nachvollziehbar dokumentiert (vgl. Abschnitt 2.1.6), indem `refine`-Beziehungen zwischen den entsprechenden Anwendungsfällen (z.B. *pick and place*) und der Anforderung gesetzt werden.

Weiterhin dargestellt ist die Dekomposition der Anforderung in Subanforderungen. Hier dargestellt als Zergliederung in die Reaktionskräfte in den drei Raumrichtungen. In Abb. 4.11 ist der Packagebrowser eines Modells für Parallelroboter gezeigt. Innerhalb des *Package* Geometrie (ein Gliederungspunkt) befindet sich die Anforderung Arbeitsraum bereitstellen mit ihrer „Tochter“ notwendige Bewegungsweite und „Enkeltochter“ notwendige Bewegungsweite in x-Richtung. Die Anforderungen auf der dritten Ebene sind die detailliertesten und sollen sich für eine quantitative Beschreibung und zielgerichtete Entwicklung eignen.

Als weitere Intermodellbeziehung wird die `allocate`-Beziehung genutzt, um Anforderungen bestimmten Systemkomponenten (`<<block>>`) zuzuweisen. Dabei wird in dem entwickelten Konzept von der in der SysML vorgesehenen Anwendung abgewichen. Dort bildet die `allocate`-Beziehung z.B. logische Systembausteine oder Aktionen auf physikalische Systembausteine ab, etwa Funktionen auf Funktionsträger. Die frühzeitige Anwendung auf Anforderungen und Systembau-

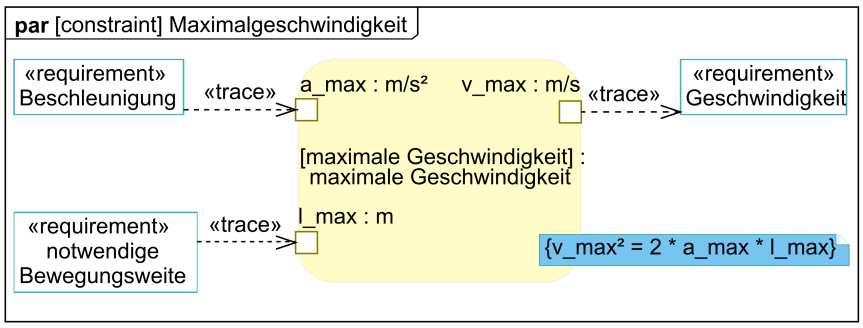


Abbildung 4.10: *Das Beispiel eines Parametrikdiagramms zeigt die quantitative Beziehung zwischen drei Anforderungen bzgl. der Maximalgeschwindigkeit am TCP eines Parallelroboters.*

Abbildung 4.11: *Darstellung der dreistufigen Anforderungshierarchie im Packagebrowser am Beispiel Parallelroboter (screenshot).*

steine bildet den Synthesegedanken ab und gewährleistet, dass diese Anforderung bei der Entwicklung der entsprechenden Komponente berücksichtigt wird. In der SysML hingegen ist die Verknüpfung von Anforderungen und Systembausteinen durch `satisfy`-Beziehungen in dem Sinne vorgesehen, dass ein Systembaustein in der Lage ist, eine Anforderung zu erfüllen. Diese Anwendung bildet lediglich den Analysegedanken ab, der z.B. bei der Überprüfung auf Änderungsauswirkungen genutzt werden kann.

4.1.7 Systemkontextmodell

Dieses Partialmodell beschreibt den Systemkontext in allen notwendigen Facetten. Die jeweils notwendige bzw. sinnvolle Tiefe der Modellierung muss durch Erfahrung der Entwickler festgelegt werden. Dazu werden auf abstrakter Ebene die in Abb. 4.1 dargestellten Aktivitäten des Bausteins „Systemkontext“ durchgeführt. Das Produkt wird segmentiert und die Segmente (z.B. Hauptbaugruppen) im Modell auf Systemebene festgehalten. Außerdem werden die jeweils notwendigen Ein- und Ausgangsgrößen bestimmt, festgehalten und damit die Beziehungen der Segmente untereinander definiert. Der spezifische Entwurf auf Subsystemebene findet zu großen Teilen in spezialisierten Modellen statt, die durch spezielle Programme unterstützt werden (z.B. CAD, FEM). Durch das übergeordnete hier beschriebene Modell sind die Schnittstellen zwischen den spezifischen Entwürfen festgelegt. Insbesondere wird dokumentiert, welche Eingabewerte ein Segment von einem anderen benötigt. Beispielsweise werden die Abmessungen von Sensoren benötigt, um sie in eine Struktur konstruktiv integrieren zu können. Darüberhinaus wird für ein Regelungskonzept die Auflösung der Sensoren benötigt. Durch die Modellierung dieser Zusammenhänge wird die Notwendigkeit zur Sy-

stemintegration auch im spezifischen Entwurf berücksichtigt. Dadurch steht die Entwicklung eines optimalen Gesamtsystems im Vordergrund und nicht die Entwicklung optimaler Subsysteme, die kombiniert ein suboptimales Gesamtsystem ergeben (vgl. Abschnitt 2.2.4).

Ein weiterer Aspekt des Systemkontextmodells ergibt sich für Weiter- oder Anpassentwicklungen: Vorhandene Modelle können hinsichtlich ihrer Schwachstellen analysiert werden und bieten damit die Ausgangspunkte für eine Weiterentwicklung. Außerdem sind sie der Benchmark an dem sich das neue Produkt messen muss.

Um eine vollständige Beschreibung des gesamten Systemkontextes realisieren zu können, besteht das Systemkontextmodell selbst aus einer Reihe unterschiedlicher „Untermodelle“, die fachbereichsspezifisch ausgearbeitet sind. Diese sind so detailliert wie jeweils notwendig modellierbar. Die „Untermodelle“ sind in der SysML teilweise bereits enthalten. Auf den ersten Blick sind sie allerdings nicht auf eine konstruktionsmethodische Entwicklung abgestimmt, da sie sich sehr stark an der Software- und Elektronikentwicklung orientieren. Dennoch können die Modellierungsmethoden hier eingesetzt werden, wenn der Konstrukteur sich etwas von eingefahrenen Wegen löst. Beispielsweise ist ein Aktivitätsdiagramm nutzbar, um Funktionsstrukturen darzustellen. Dabei müssen Flüsse über Parameter in die Aktivitäten geführt werden. Den Parametern werden Typen zugeordnet, so dass die übliche Beschreibung durch Stoff-, Informations- und Energieflüsse möglich ist. Die Parameter können darüberhinaus mit immer konkreteren Typen (z.B. Kraft und Newton) belegt werden. Weiterhin ist es problemlos möglich „Hauptfunktionen“ sukzessive zu dekomponieren. In den üblichen Werkzeugen öffnet sich auf Doppelklick ein neues Diagramm, welches das „Innere“ der Hauptfunktion darstellt. Wie im vorherigen Abschnitt beschrieben, können Aktivitäten (Funktionen) schließlich Systembausteinen (Funktionsträgern) zugeordnet werden.

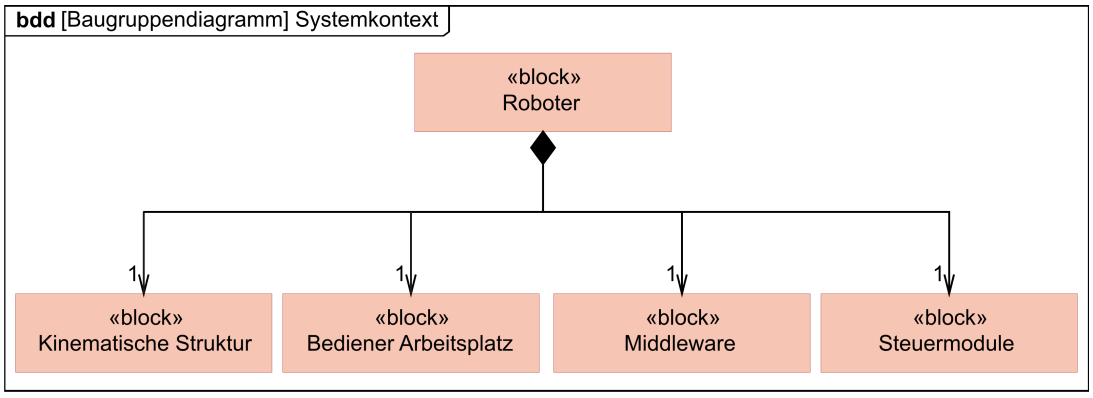


Abbildung 4.12: *Diagramm der Hauptbaugruppen aus denen sich das Robotersystem zusammensetzt.*

Die Module oder Komponenten werden (wie bereits für das Produktumgebungsmodell in Abschnitt 4.1.4) durch `<<block>>`-Objekte beschrieben. Abb. 4.12 zeigt ein *Block Definition Diagram* `[Baugruppendiagramm] Systemkontext` mit den Hauptbaugruppen des Robotersystems. Dabei handelt es sich um physische (z.B. Kinematische Struktur) und softwaretechnisch umgesetzte Baugruppen (z.B. Middleware). Die dargestellten Beziehungen sind Kompositionen, d.h. die Teile (Hauptbaugruppen) existieren nur für das Robotersystem.

Die Hauptbaugruppen werden je nach Bedarf weiter aufgegliedert. Es sind in Abb. 4.13 beispielhaft die Unterbaugruppen der Hauptbaugruppe „Kinematische Struktur“ dargestellt. Es ist zu erkennen, dass eine kinematische Struktur aus zwei bis sechs kinematischen Ketten aufgebaut werden kann. Weiterhin sind Schnittstellen zwischen den Unterbaugruppen als Assoziationen abgebildet. Die zwei bis sechs kinematischen Ketten sind auf der einen Seite am Gestell und auf der anderen Seite an der Arbeitsplattform angebunden. Zwischen Arbeitsplattform und Effektor können sich zudem zusätzliche Aktoren befinden.

Bei der Darstellung in einem *Block Definition Diagram* ist es möglich, Blöcke durch Vektorgraphiken zu visualisieren. Dadurch wird eine in der Ebene grob maßstäbliche Repräsentation der Module und ihres Zusammenwirkens möglich. *Blees* beschreibt eine ähnliche Darstellung als MIG [Ble08b]. Die Umsetzung des MIG in SysML erlaubt eine einfachere rechnerunterstützte Handhabung des Kon-

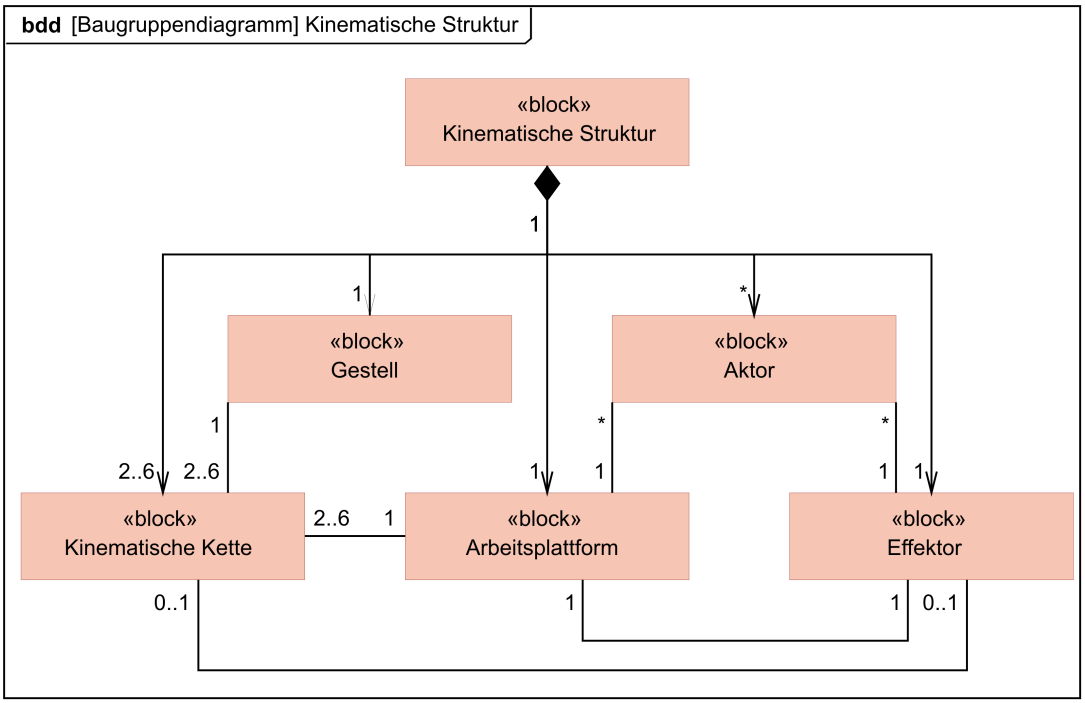


Abbildung 4.13: *Diagramm der Hauptbaugruppe „Kinematische Struktur“ des Robotersystems.*

zepts und eine verbesserte objektorientierte Beschreibung und Wiederverwendung der Module und Schnittstellen.

4.1.8 Testmodell

Das Testmodell soll es erlauben, die Zielerfüllung frühzeitig zu überprüfen und somit ein qualitativ hochwertiges, den Kundenwünschen entsprechendes und die gesetzten Ziele erreichendes Produkt zu entwickeln (vgl. Abschnitt 2.1.4). Dazu wird beschrieben, wie das System bzw. einzelne Systemkomponenten getestet werden sollen (vgl. Abb. 4.1). Darüberhinaus eignet es sich das Entwicklungsrisiko des Projektes zu beurteilen.

Das Testmodell besteht im Wesentlichen aus drei unterschiedlichen Objektarten: Den Testkriterien, den Testfällen und den Testwerkzeugen.

Die Testkriterien werden direkt aus den Anforderungen entwickelt und sind somit über `satisfy`-Beziehungen mit dem Anforderungsmodell verbunden. Falls das Testteam nicht in der Lage ist, zu einer Anforderung ein entsprechend überprüfbares Testkriterium zu definieren, muss die Anforderung überdacht werden. Entweder war die Anforderung nicht präzise genug formuliert oder es gibt keine Möglichkeit ihre Erfüllung sinnvoll zu überprüfen. Falls letzteres der Fall ist, muss die Anforderung aus dem Anforderungskollektiv entfernt werden.

Testkriterien werden im Modell zunächst als `<<requirement>>` erzeugt und anschließend mit dem neu erarbeiteten Stereotyp `<<testCriterion>>` versehen. Dadurch erhält ein Testkriterium die folgenden Merkmale:

- `Was wird getestet`: Dieses Merkmal beschreibt, welche Art von Eigenschaften des Testobjekts getestet wird (Funktion, Struktur oder Verhalten).

- `Direktes Testen`: Das direkte Testen beschreibt, ob Testkriterien direkt getestet werden können bzw. sollen oder über Zwischengrößen indirekt getestet werden, z.B. Kraft über die Auslenkung einer Feder.

- `Fachbereich`: An der Entwicklung komplexer Produkte sind häufig eine Reihe unterschiedlicher technischer Fachbereiche beteiligt. Die Testkriterien können aus unterschiedlichen Fachbereichen stammen und erfordern ggf. spezielle Testverfahren (z.B. Schwingungstest, Software-in-the-Loop).

- `Risiko`: Bei der Produktentwicklung bringen unterschiedliche Eigenschaften und damit die jeweiligen Testkriterien unterschiedliche Risiken mit sich. Das Risiko wird hier in Anlehnung an die FMEA als Risikoprioritätszahl (RPZ) aufgefasst. Es setzt sich multiplikativ aus folgenden Größen zusammen:

  - `Auftretenswahrscheinlichkeit` A: Beschreibt die geschätzte Wahrscheinlichkeit, dass das Testkriterium nicht erfüllt wird.

- `Bedeutung` B: Je größer bzw. schwerwiegender die Folgen des Auftretens eines Fehlers sind, desto höher ist die Bedeutung des Testkriteriums.
- `Entdeckungswahrscheinlichkeit` E: Dieses Merkmal beschreibt, wie wahrscheinlich es ist, dass ein Fehler dieses Testkriteriums entdeckt wird.

Zusätzlich sind die bereits beschriebenen *Tag Definitions* `author`, `responsable` und `state` enthalten.

Wie beschrieben, müssen alle Anforderungen testbar, d.h. durch Testkriterien abgedeckt sein. Es handelt sich also um eine Erfüllungsbeziehung, die im Modell durch `satisfy` abgebildet wird.

Testkriterien werden in Testfällen getestet. Zwischen Testfällen und Testkriterien wird die SysML-Beziehung `verify` gesetzt, um anzugeben, dass ein formalisierter Testfall ein bestimmtes Testkriterium überprüft.

Testfälle werden als *Use Cases* erzeugt und anschließend mit dem SysML-Stereotyp `<<testCase>>` versehen.

Für ein systematisches Vorgehen können Testfälle strukturiert werden. Es gibt eine Anzahl unterschiedlicher Klassifizierungen innerhalb der verschiedenen Fachbereiche (z.B. *code and unit test*, *module test*, *system test* in der Softwareentwicklung), die sich zum Teil überschneiden. Eine einheitliche fachbereichsübergreifende Systematik ist bisher nicht bekannt. In Abb. 4.14 ist ein Ausschnitt des Konstruktionskataloges gezeigt, der zur systematischen Beschreibung von Testfällen für die Parallelroboterentwicklung erarbeitet wurde. Er liegt derzeit als Excelarbeitsmappe vor und enthält umfangreiche Beschreibungen und Erläuterungen, auf die mittels Hyperlinks zugegriffen werden kann. Der Gliederungsteil berücksichtigt die drei im folgenden beschriebenen Merkmale mit ihren jeweiligen Merkmalsausprägungen. Hieraus ergeben sich 36 unterschiedliche Klassen von Testfällen. Dem vorhandenen SysML-Stereotyp `<<testCase>>` wurden die im Folgenden vorgestellten Merkmale als neue *tag definitions* hinzugefügt.

- 1.1 Was wird getestet?
  Das erste Gliederungsmerkmal (vgl. Tabelle 4.1) beschreibt, in welchem abstrakten Bereich getestet werden soll. Die Institute of Electrical and Elec-

Tabelle 4.1: *Gliederungsmerkmale und ihre Merkmalsausprägungen für Testfälle.*

Was wird getestet?	**Testobjekt**	**Aggregationsstufe**
Funktion	*abstrakt*	*Bauelement*
Struktur	*virtuell*	*Baugruppe*
Verhalten	*real*	*Maschine*
		Anlage

Gliederungsteil			Hauptteil			Zugriffsteil												
1.1	1.2	1.3				1.4	1.5	1.6	1.7	1.8	1.9	1.10	1.11	1.12	1.13	1.14	1.15	1.16
Was wird getestet?	Testobjekt	Aggregationsstufe	Bezeichnung	Kurzbeschreibung		Dynamik	Kombiniertes Testen	Vorwissen	Wer testet?	Phase im PEP	Testkreislauf	Testumgebung	Schnittstellen	Automatisierbarkeit	Zeitraum	Aufwand	Übertragbarkeit	Abdeckung
1	2	3	1	2	Nr.	1	2	3	4	5	6	7	8	9	10	11	12	13
Funktion	abstrakt	Bauelement	Konzepttest	Beurteilen anhand allgemeiner Informationen…	1	3	1 + 2 + 3	1	1	1	1 + 2	1	1	2 + 3	3	3	3	3
		Baugruppe		Beurteilen anhand allgemeiner Informationen…	2	3	1 + 2 + 3	1	1	1	1 + 2	1	1	2 + 3	3	3	3	3
		Maschine		Beurteilen anhand allgemeiner Informationen…	3	3	1 + 2 + 3	1	1	1	1 + 2	1	1	2 + 3	3	3	3	3
		Anlage		Beurteilen anhand allgemeiner Informationen…	4	3	1 + 2 + 3	1	1	1	1 + 2	1	1	2 + 3	3	3	3	3
	virtuell	Bauelement		Beurteilen anhand einer Simulation…	5	1	1 + 2 + 3	2	1	2	1 + 2	1 + 2 + 3	1 + 2 + 3	2	2	2	1	1

Abbildung 4.14: *Ausschnitt aus einem Konstruktionskatalog zur systematischen Einteilung von Testfällen. Die Ziffern im Zugriffsteil stehen für Ausprägungen der jeweiligen Zugriffsmerkmale.*

tronics Engineers (IEEE) schlägt für Softwarewerkzeuge, die einen Informationsinhalt repräsentieren sollen, die folgende Einteilung vor [IEE98b, S. 5]: den objektorientierten Ansatz (Attribute und Dienstleistungen von „realen“ Objekten), den prozessbasierten Ansatz (Hierarchien von Funktionen, die über Datenflüsse miteinander kommunizieren) und den das Verhalten beschreibenden Ansatz (externes Systemverhalten in Form von abstrakten Notationen, mathematischen Funktionen oder Zustandsautomaten). Als fachbereichsübergreifende Einteilung wird allgemein nach Funktion, Struktur und Verhalten unterschieden:

– Die Funktion beschreibt die Überführung von Eingangsgrößen in Ausgangsgrößen unter Berücksichtigung von Parametern und Störgrößen, ggf. in einer zeitlichen und logischen Abfolge. Beispielsweise hat die Funktion *Wagen heben* die Eingangsgrößen Fahrzeuggewicht und Handkraft und die Ausgangsgröße Abstand zwischen Straße und Auto. Zu berücksichtigende Parameter sind die Steigung im Bewegungsgewinde. Ein Beispiel für eine Störgröße ist die Außentemperatur.
Mathematisch handelt es sich um eine Funktion von einem oder meh-

reren Produktparametern:

$$F(x) = 2\,\frac{N}{m} \cdot x \tag{4.1}$$

Ein Testkriterium überprüft hier die Funktionserfüllung, d.h. ob bei gegebenen Eingangsgrößen die gewünschten Ausgangsgrößen erreicht werden.

- Die Testkriterien der Merkmalsausprägung Struktur berücksichtigen beispielsweise geometrische Randbedingungen (z.B. verfügbare Bauräume) oder das Einhalten einer gewünschten Architektur (z.B. Vorhandensein notwendiger Schnittstellen). Für ein Robotersystem wird hier z.B. überprüft, ob mit den gewählten Gliedlängen bei der festgelegten Anordnung der kinematischen Ketten die gewünschte Arbeitsraumgröße erreicht wird.
- Das Verhalten beschreibt die dynamische Überführung von Eingangsgrößen in Ausgangsgrößen (im Allgemeinen Zustandsgrößen). In informationstechnischen Systemen wird durch ein Ereignis der dynamische Übergang zwischen Zuständen ausgelöst. Beispielsweise können Roboter durch Aktionsprimitive in einigen Bereichen autonom gesteuert werden, wodurch z.B. eine vereinfachte und robustere Roboterprogrammierung ermöglicht wird. [Fin05]
  Das dynamische Verhalten kann getestet werden, indem z.B. ein Soll-Ist-Abgleich über einen bestimmten Zeitraum durchgeführt wird. Die Analyse des Einschwingvorgangs einer drehelastischen Schaltkupplung ist ein Beispiel hierfür.

- 1.2 Testobjekt
  Das zweite Gliederungsmerkmal beschreibt das Abstraktionsniveau auf dem das Testobjekt zum Zeitpunkt des Tests vorliegt: abstrakt (z.B. als Funktionsstruktur, Skizze), virtuell (z.B. als CAD-Modell, DMU) oder real (z.B. als Rapid Prototyping (RP) oder konventionell gefertigter Prototyp). Dieses Merkmal beschreibt indirekt ab welchem Zeitpunkt Testfälle dieser Testklasse angewendet werden können, da virtuelle und reale Testobjekte erst in mittleren und späten Phasen der Entwicklung vorliegen. Für die Verwendung von RP-Objekten existieren neuerdings spezielle Konstruktionskataloge, die hier weit tiefer ins Detail gehen [Kir09].

- 1.3 Aggregationsstufe
  Die Aggregationsstufe beschreibt das Komplexitätsniveau des Testobjekts abgestuft von Bauelement bis Anlage. Der Hintergrund ist, dass auf Ebene von Bauelementen wesentlich feingranularer getestet werden kann, während auf Ebene der Anlage häufig nur Integrationstests durchgeführt werden können. Beispielsweise ist eine Finite Elemente Analyse für größere Bereiche

eines komplexen Bauteils oder gar ganze Maschinen mit heutiger Rechenleistung nicht zu realisieren.

Der Zugriffsteil enthält weitere die Testfälle beschreibende Merkmale. Diese eignen sich auf Grund von gegenseitigen Abhängigkeiten nicht als Gliederungsmerkmale (z.B. beeinflusst die Abdeckung den Aufwand und die Übertragbarkeit). Außerdem können die vergleichsweise allgemeinen Testfallklassen nicht immer eindeutig mit einer Merkmalsausprägung belegt werden. Die Zugiffsmerkmale eignen sich, um die Testfallklassen weiter zu charakterisieren und systematisch herauszufinden, welche Klasse sich für welche Testarten besonders gut oder schlecht eignet. Beispielsweise sind abstrakte Testfälle nur schwer automatisierbar. Eine definitive Entscheidung über die Automatisierbarkeit wird dann für einen konkreten Testfall gefällt. Im Folgenden werden die Zugriffsmerkmale genauer beschrieben:

- 1.4 Dynamik
  Tests können statisch (bzw. quasistatisch) oder dynamisch durchgeführt werden.

- 1.5 Kombiniertes Testen
  Das kombinierte Testen beschreibt, ob Testkriterien einzeln oder in Kombination getestet werden können bzw. müssen. Außerdem beschreibt es, ob ein Test in der Lage ist Kombinationen von Testkriterien zu testen. Beispielsweise können maximale Abmessungen (z.B. „Packmaß“) einzeln für die drei Raumrichtungen gemessen und jeweils mit den Sollwerten verglichen werden. Wird der Gegenstand in eine Form gelegt, so kann das Einhalten der maximalen Abmessungen kombiniert überprüft werden (vgl. Kontrolle der Größe des Handgepäcks am Flughafenschalter). Im kombinierten Fall ist das Ergebnis der Überprüfung lediglich positiv oder negativ. Werden die Testkriterien einzeln getestet sind darüberhinaus Aussagen über den Abstand zur Zielerreichung möglich (z.B. der Koffer ist genau 2 cm zu breit).

- 1.6 Vorwissen
  Dieses Merkmal beschreibt, wie viel bereits vor dem Test über Systeminterna bekannt sein muss. Die Extreme bilden *black-box* und *white-box* Tests. Bei ersteren sind lediglich die Schnittstellen bekannt (z.B. hardware-in-the-loop), bei letzteren ist der komplette Systemaufbau bekannt (z.B. FEM).

- 1.7 Wer testet?
  Das entwickelte Produkt bzw. dessen Komponenten werden im Verlauf der Produktentwicklung von unterschiedlichen Personengruppen getestet (nach [IEE08, S. 26]). Es wird unterschieden zwischen einem Test durch den Entwickler selbst, durch ein Testteam oder durch den Nutzer. Dabei vergrößert sich der Regelkreis und die Reaktionszeit auf die Testergebnisse. Der Entwickler pflegt durch eigene Testergebnisse induzierte Änderungen zeitnah

(d.h. während der Entwicklung) in das Produkt ein. Änderungen, die erst durch Tests des Nutzers im Betrieb bzw. bei Inbetriebnahme (Abnahmetests) induziert werden, sind sehr aufwändig. In der Regel kann hier nur eine teure Fehlerkorrektur vorgenommen werden, z.B. indem adaptronische Komponenten eingesetzt werden, die bei der Entwicklung nicht berücksichtigte Schwingungen kompensieren.

- 1.8 Phase im PEP
  Der Produktentwicklungsprozess lässt sich in verschiedene Phasen unterteilen. In den verschiedenen Phasen werden unterschiedliche Testkriterien getestet und unterschiedliche Arten von Tests sind notwendig bzw. möglich. Dieses Merkmal beschreibt, in welcher Phase der Test durchgeführt wird.

- 1.9 Testkreislauf
  Das Objekt, welches getestet werden soll, wird in eine Testumgebung eingebettet. Die Testumgebung stellt die Eingabewerte an das Testobjekt bereit. Bei einem geschlossenen Testkreislauf werden die Ausgabedaten des Testobjekts an die Testumgebung zurückgegeben. Innerhalb der Testumgebung werden aus den Rückmeldungen des Testobjekts neue Eingabedaten entwickelt (vgl. [Kno07]). Ein offener Testkreislauf liefert lediglich die zuvor festgelegten Eingabedaten und zeichnet die Ausgabedaten auf. Es handelt sich demnach nicht um einen Kreislauf im eigentlichen Sinn.

- 1.10 Testumgebung
  Das Testobjekt wird zum Testen in eine Testumgebung eingebettet. Diese Testumgebung kann—wie das Testobjekt selbst—in unterschiedlicher Konkretisierung vorhanden sein.

- 1.11 Schnittstellen
  Beschreibt die Art der Schnittstellen zwischen dem Testobjekt und der Testumgebung. Die Schnittstellen können abstrakt oder vollständig real sein. Zwischen beiden Extremen liegen sogenannte *Stubs*. Diese stellen die Eingangswerte bereit und nehmen Ausgangswerte an. Die übrige Testumgebung liegt häufig auf einer anderen Abstraktionsebene vor. Beispielsweise werden reale Computer-Steckkarten an reale Steckplätze angeschlossen, hinter denen verschiedene Systeme und Systemzustände mit einer virtualisierten Testumgebung simuliert werden.

- 1.12 Automatisierbarkeit
  Die Durchführung von Tests ist im Allgemeinen (zeit-)aufwändig, daher ist man bestrebt, möglichst große Anteile der Testprozeduren zu automatisieren. Dieses Merkmal beschreibt, ob ein Test automatisierbar oder teilautomatisierbar ist.

- 1.13 Zeitraum
  Die Tests können über unterschiedlich lange Zeiträume durchgeführt werden. Beispielsweise ist für einen Gesamtfahrzeugkorrosionstests ein vergleichsweise großer Zeitraum zu veranschlagen, während für den Funktionstest einer Prozedur im Programmcode ein im Allgemeinen vergleichsweise kurzer Zeitraum benötigt wird. Der Durchlauf von schlecht oder nicht konvergierende Algorithmen kann jedoch auch einen großen Zeitraum in Anspruch nehmen, falls keine geeigneten Abbruchkriterien festgelegt wurden.

- 1.14 Aufwand
  Dieses Merkmal beschreibt qualitativ den Aufwand, der erfolgen muss, um einen Test durchzuführen. Der Aufwand berücksichtigt dabei Vorarbeiten (z.B. Teststand bauen), die eigentliche Testdurchführung (Daten aufzeichnen) und Nacharbeiten (z.B. Auswerten/Interpretieren der Aufzeichnungen). Es wird hier nicht einzeln zwischen zeitlichem (Arbeitskraft) und monetärem (z.B. Investition in neue Messausrüstung) Aufwand unterschieden.

- 1.15 Übertragbarkeit
  Dieses Merkmal berücksichtigt die Übertragbarkeit der Testergebnisse auf die realen Einsatzbedingungen des zu testenden Systems. Jeder Test unterliegt bestimmten Randbedingungen und bildet die Realität vereinfacht ab. Sind die verwendeten Modelle stark vereinfacht (z.B. in frühen Entwicklungsphasen), kann ein Test oftmals nur eine Reihung von besseren gegenüber schlechteren Lösungsideen liefern.

- 1.16 Abdeckung
  Die Abdeckung beschreibt, wie groß der durch den Test abdeckbare Zustandsraum im Vergleich zum möglichen Zustandsraum ist. Bei netzartig aufgebauten Softwaresystemen, in denen Zustände mehrfach durchlaufen werden können, ist eine vollständige Abdeckung z.B. kaum möglich. Fehler, die nur bei unwahrscheinlicher und vom Programmierer eigentlich nicht vorgesehener Abfolge von Zuständen auftreten, werden selten gefunden. Gleiches gilt für Versuche mit zahlreichen Einflussfaktoren, bei denen die Kombinatorik eine große (bei analogen Systemen eine nahezu unendliche) Anzahl möglicher Parameterkollektive vorgibt. Hier wird versucht z.B. durch statistische Versuchsplanung die wahrscheinlichsten Zustände abzudecken.

Als dritte Komponente werden Testwerkzeuge als Blöcke im Modell dargestellt. Sie dienen im Wesentlichen als Wissensspeicher, um festzuhalten, welche Eigenschaften ein bestimmtes Testwerkzeug aufweist. Beispielsweise bietet Unigraphics als CAD-Modellierer eine Kollisionskontrolle der montierten Baugruppe in festgelegten Konfigurationen und Posen. Die Testwerkzeuge sind geeignet die Überprüfung von Testkriterien bestimmter Testfallklassen zu unterstützen.

Abb. 4.15 zeigt auf der linken Seite die Testfallklasse 7 als `<<TestCase>>`.

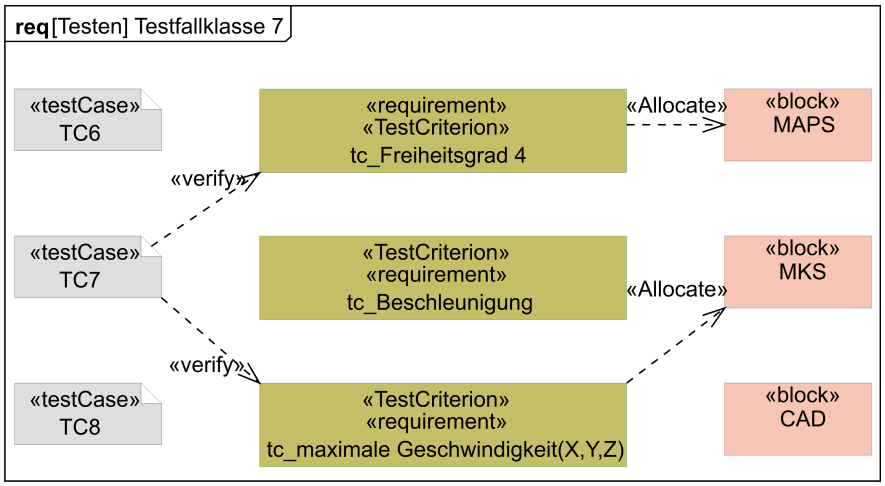


Abbildung 4.15: *Ausschnitt aus einem Anforderungsdiagramm zur Abbildung von Testfällen, Testkriterien und Testwerkzeugen.*

In dieser Klasse wird die Funktion auf Maschinenebene virtuell getestet. Nach Abb. 4.14 ist die Übertragbarkeit der Ergebnisse von Tests dieser Klasse bei überschaubarem Aufwand vergleichsweise gut anzunehmen. Rechts neben den Testfällen sind einige Testkriterien dargestellt, die—gekennzeichnet durch `verify`-Beziehungen—durch Testfälle dieser Testfallklasse getestet werden können. Beispielsweise soll hier überprüft werden, ob das Robotersystem den angestrebten Freiheitsgrad 4 am Tool Center Point (TCP) bereitstellen wird. Auf der rechten Seite finden sich schließlich die hierfür in Frage kommenden Testwerkzeuge. Sie sind mit `allocate`-Beziehungen den entsprechenden Testkriterien zugeordnet. Es ist zu erkennen, dass der Freiheitsgrad mit Hilfe des Werkzeugs *MAPS* überprüft wird. Es handelt sich dabei um ein in Matlab programmiertes Werkzeug zur kinematischen Analyse von Parallelstrukturen [Fri01].

4.2 Arbeiten mit dem Modell

Mit den in Abschnitt 3.4 beschriebenen Möglichkeiten zur Auswertung wird das Modell ständig analysiert und verbessert. Im Folgenden werden die konkreten Auswertemechanismen vorgestellt. Diese reichen von der Sichtenbildung in Listen und Diagrammen über verschiedene Matrizen bis zu kennzahlbasierten Auswertungen in Tabellen bzw. ihrer Visualisierung in verschiedenen Grafiken.

4.2.1 Bildung von spezifischen Sichten

Die Sichtenbildung erlaubt es, Sammlungen von Objekten nach bestimmten Merkmalsausprägungen und Abhängigkeiten zu filtern. Insbesondere lassen sich Anforderungslisten bestimmter Sichtweisen erstellen, die für eine aktuelle Aufgabe

sinnvoll erscheinen. Um den Objektumfang einzugrenzen werden dabei ggf. Filter nach mehreren Merkmalsausprägungen gleichzeitig eingesetzt.

In der Phase der Aufgabenklärung lässt sich ein Benutzer beispielsweise eine Liste aller Objekte ausgeben, für die er die Verantwortung trägt. Dabei muss er nicht notwendigerweise Autor der Anforderungen sein. Anschließend filtert er die Liste nach Einträgen, die weitere Eingaben erfordern (*Status*: `created`). Nun arbeitet der Benutzer systematisch die Anforderungen seines Verantwortungsbereiches ab.

Während der Entwicklung lässt sich ein Komponentenentwickler beispielsweise eine Anforderungsliste ausgeben, die sämtliche Anforderungen einer Komponente betreffen (modelliert durch eine `allocate`-Beziehung). Zusätzlich filtert er hinsichtlich des von ihm vertretenen Fachbereichs und sortiert Wünsche aus der Liste aus. Damit hat er für die Konzeptentwicklung ein überschaubares Anforderungskollektiv vorliegen.

Ein Testfallentwickler lässt sich z.B. Anforderungen ausgeben, für die bisher noch keine `satisfy`-Beziehung zu Testkriterien gesetzt wurden. Zudem lässt er sich die Anwendungsfälle anzeigen, die mit der Anforderung in Beziehung stehen. Diese geben ihm ggf. Hinweise darauf, wie ein sinnvoller Testfall für das zu entwickelnde Testkriterium beschrieben werden kann.

Abgesehen von den Anforderungslisten lassen sich auch spezifische Diagramme erstellen, die ein bestimmtes Thema repräsentieren. Insbesondere eignen sich Diagramme, um eine gewisse Problemstellung mit Beteiligung unterschiedlicher Fachbereiche zu diskutieren (vgl. Abschnitt 2.2.5). Beispielsweise wenn die Gestaltung einer Komponente (z.B. Gelenkwinkelsensor einer kinematischen Struktur [Fra08]) großen Einfluss auf viele verschiedene Systemeigenschaften unterschiedlicher Bereiche hat (z.B. Gestaltung der Gelenke, Kalibrierung, Dynamik, Regelung).

4.2.2 Stakeholder-Anwendungsfälle-Matrix

Die Erstellung der Auswertematrizen erfolgt durch Makroprogrammierung in Microsoft Excel (Visual Basic for Applications (VBA)). Es wird auf die Objekte des zuletzt oder derzeit geöffneten Modells zugegriffen. Dazu wird das Modell zunächst als Datenbasis geladen. Anschließend werden in Microsoft Excel über das in Abb. 4.16 abgebildete Formularfenster die gewünschten Module des Programmcodes gestartet. Dazu werden in den *checkboxes* Haken für jede gewünschte Matrix gesetzt. Für die später noch beschriebene Anforderungen-Anforderungen-Matrix existieren weitere Optionen. Das Ergebnis ist eine Excel Arbeitsmappe mit den entsprechenden Auswertematrizen als Tabellenblätter.

Ein Beispiel für eine Stakeholder-Anwendungsfälle-Matrix ist in Abb. 4.17 dargestellt. Zuerst werden durch mehrere Schleifen die Stakeholder entsprechend ihrer Ebene in den ersten fünf Kopfspalten eingeordnet (vgl. Abschnitt 4.1.5). Für Objekte der unteren Ebenen werden die darüberliegenden Objekte durch

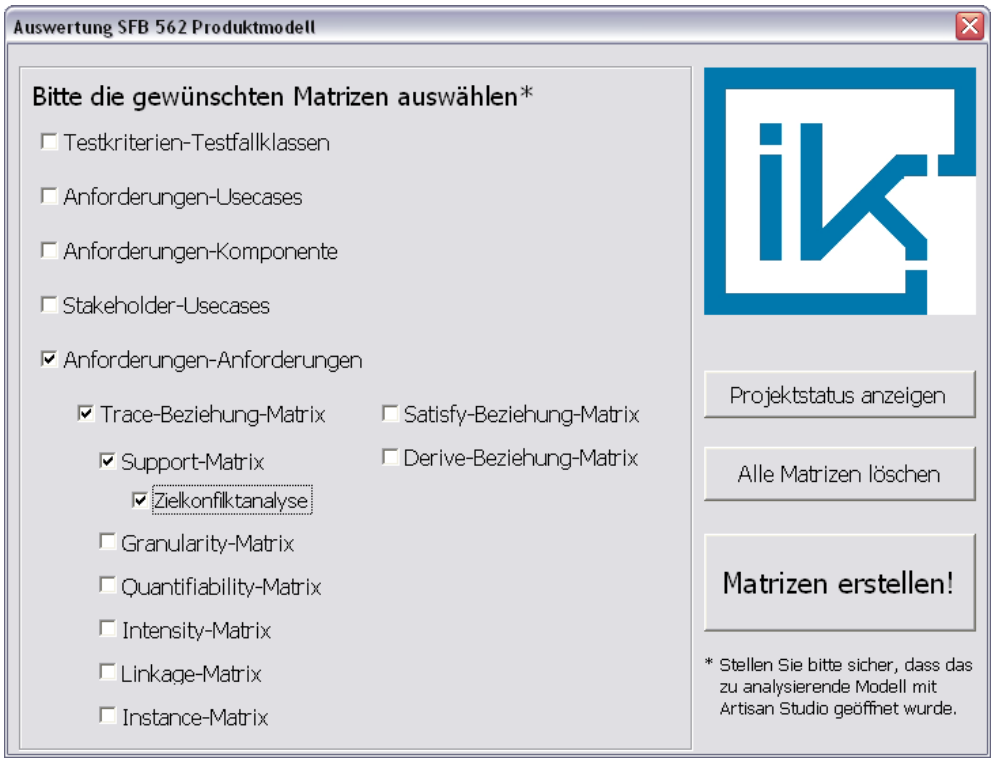


Abbildung 4.16: *Formularfenster in Microsoft Excel zum automatischen Erstellen der gewünschten Auswertematrizen.*

Generalisierungsbeziehungen identifiziert und in die Zelle jeweils links daneben geschrieben (z.B. Hersteller als Generalisierung von Entwickler). So ist es möglich die Matrix nach Einträgen der Ebenen zu strukturieren und z.B. alle Stakeholder des Herstellers bis zu einer bestimmten Ebene zu betrachten. Durch eine Gegenüberstellung der natürlichen Personen zweier Rollen der oberen Ebenen (z.B. Entwickler und Fertiger) werden Personen identifiziert, die in den gleichen Anwendungsfällen involviert sind und sich entsprechend absprechen müssen.

In den Kopfzeilen wird der Produktlebenslauf abgebildet (vgl. Abschnitt 4.1.3). Dazu werden sechs Zeilen verwendet. Die oberen fünf enthalten die Namen der Packages in denen die Anwendungsfälle einsortiert sind. Die Anwendungsfälle selbst stehen in der sechsten Kopfzeile. Dadurch ist es möglich die Anwendungsfälle zu sortieren und bspw. den zeitlichen Verlauf im Produktlebenslauf von links nach rechts abzubilden (vgl. Abb. 4.5). Außerdem ist es möglich, die für eine bestimmte Fragestellung interessierenden Anwendungsfälle zweier Lebenslaufphasen (z.B. Entwicklung und Fertigung) einander gegenüberzustellen.

Durch Kopfzeilen und -spalten werden Zellen aufgespannt. Ist ein Stakeholder an einem Anwendungsfall beteiligt, so wird die entsprechende Zelle mit „ja“ markiert. Dazu wird durch das Makro für jede Zelle überprüft, ob der jeweilige Anwendungsfall im Modell eine Beziehung zum entsprechenden Stakeholder

					Lebenslaufphasen	Vor der Nutzung			
						Entwicklung		Fertigung	
						Systemkontextentwicklung			
						Gestaltung			
					Anwendungsfälle	Kinematik festlegen	Komponenten festlegen	Fertigungs-planung	Eigen-fertigung
Stakeholder									
						Ja	Ja	Ja	Ja
Hersteller						Ja	Ja	Ja	Ja
	Entwickler					Ja	Ja	Ja	
		Entwickler MB	Konstrukteur			Ja	Ja	Ja	
		Entwickler System					Ja	Ja	
			Anforderungsanalytike		Carsten Stechert				
	Fertiger						Ja	Ja	Ja
		Arbeitsvorbereitung						Ja	Ja
		Fertigungsabteilung					Ja	Ja	Ja
		Montageplaner					Ja		
	Qualitäts-manager							Ja	Ja
		Qualitätskontrolleur						Ja	Ja
	Verwaltung					Ja	Ja	Ja	Ja
		Manager	Projektleiter		Hans-Joachim Franke	Ja	Ja		
					Thomas Vietor	Ja	Ja		
Kunde	Anwender								
	Betreiber								
	Käufer								
	Produktent-wickler des Kunden	Chemiker							
		Kosmetikentwickler							
		Lebensmitteltechnologe							

Abbildung 4.17: *Ausschnitt aus einer Stakeholder-Anwendungsfälle-Matrix für Parallelroboter.*

aufweist. Beispielsweise ist der Konstrukteur wesentlich daran beteiligt die Komponenten festzulegen. Aber auch die Fertigungsabteilung wird an der Komponentenfestlegung beteiligt, um z.B. die Herstellbarkeit zu gewährleisten.

Aus der Matrix lassen sich nun verschiedene Informationen entnehmen. Zunächst zeigt sie schnell und intuitiv welche Stakeholder über große Bereiche des Produktlebenslaufs involviert sind. Dabei lässt sich zwischen Rollen und Personen (Ebene 5) unterscheiden. Dominante Rollen sind z.B. Entwickler, Testteam, Kostenanalytiker und Projektleiter. Da eine natürliche Person mehrere Rollen gleichzeitig übernehmen kann (z.B. Aerodynamiker, Mechanikentwickler und CAD-Modellierer) und ggf. nur Teilaspekte einer Rolle ausfüllt (z.B. Entwicklung und Gestaltung der Außenhaut des rechten Fahrzeugaußenspiegels) sind die dominanten Personen häufig von den Rollen unterschiedlich und stark projektspezifisch.

Zur Gewährleistung einer guten und effizienten Zusammenarbeit, insbesondere bei verteilter Produktentwicklung (vgl. Abschnitt 2.2), müssen mindestens die dominanten Stakeholder in regem Austausch miteinander stehen. Auf Basis der Matrix wird nun eine Projektplanung möglich, die einzelne Personen bestimmten Rollen und damit Aktivitäten im Produktlebenslauf zuordnet. Dadurch wird deutlich, welche Personen zusammenarbeiten müssen. Geeignete Kommunikationsmedien und -wege werden frühzeitig festgelegt.

Eine mögliche Strategie ist es, Personen festzulegen, die als dominante Stakeholder ggf. mehrere Rollen wahrnehmen und so als Wissensträger an allen Phasen

des Produktlebens teilhaben. Für die Anforderungserfassung hat das den Vorteil, dass die Anzahl der zu befragenden Personen relativ klein gehalten werden kann, ohne die Aussagekraft des Anforderungskollektivs stark zu beeinträchtigen. Hierdurch wird die Anforderungserfassung weniger komplex und zeitintensiv und somit effizienter gestaltet.

Darüberhinaus können mit Hilfe der Matrix Rollen identifiziert werden, die für eine Klärung der Aufgabenstellung notwendig aber noch nicht mit Personen belegt sind. Hier ergeben sich Ansatzpunkte, um z.B. beim Kunden oder Lieferanten Kontaktdaten von notwendigen Ansprechpartnern zu erfragen und diese Personen ggf. in die Entwicklung einzubinden.

4.2.3 Anforderungen-Anwendungsfälle-Matrix

Die Erstellung der Auswertematrix zur Gegenüberstellung von Anforderungen und Anwendungsfällen erfolgt wieder durch Makroprogrammierung in Microsoft Excel. Die automatische Erstellung der Matrix wird ebenfalls mit dem in Abb. 4.16 abgebildeten Formular ausgelöst.

Abb. 4.18 zeigt ein Beispiel einer Anforderungen-Anwendungsfälle-Matrix. Die Anwendungsfälle werden ebenfalls nach Packages strukturiert in sechs Kopfzeilen abgebildet (vgl. Abschnitt 4.2.2). Der abgebildete Ausschnitt zeigt die Lebenslaufphasen *Verwendung* und *Wartung*, die beide während der Nutzung eintreten. Für die *Verwendung* werden zwei verschiedene Anwendungsfälle aus dem Bereich *Aufgaben* und dort der *Branche* der *Lebensmittelindustrie* betrachtet. Der erste beschäftigt sich mit der Handhabung von kleinen Gebäckstücken (*„Sortieren von Muffins“*) und der zweite mit dem Sortieren von Getränkeflaschen. Der Anwendungsfall aus der Wartung berücksichtigt das Tauschen des Werkzeugs.

Die Anforderungen werden in drei Ebenen in den Kopfspalten abgelegt (vgl.

			Während der Nutzung		
		Lebenslaufphasen	Verwendung		Wartung
			Aufgaben		
			Branchen		
			Lebensmittelindustrie		
		Anwendungsfälle	Sortieren von Muffins	Sortieren von Getränkeflaschen	Tauschen des Werkzeugs
Anforderungen					
Arbeitsraum breitstellen			Refine	Refine	Refine
	notwendige Bewegungsweite		Refine	Refine	
		...in x-Richtung	Refine	Refine	
		...in y-Richtung	Refine	Refine	
		...in z-Richtung	Refine	Refine	
	notwendige Arbeitsraumgeometrie				Refine
		Breite			Refine
		Höhe			Refine
		Länge			Refine

Abbildung 4.18: *Ausschnitt aus einer Anforderungen-Anwendungsfälle-Matrix für Parallelroboter.*

Abschnitt 4.1.6). Die Ebenen der Anforderungen bilden dabei die Dekomposition ggf. mit einhergehender Spezialisierung der Anforderungen ab und werden innerhalb des Modells durch Dekompositionsbeziehungen (Eltern-Kind-Beziehung) abgebildet. Im abgebildeten Beispiel für Parallelroboter wird die Anforderung „*Arbeitsraum bereitstellen*“ zunächst in „*Notwendige Bewegungsweite sicherstellen*“ und dann in „*Notwendige Bewegungsweite in x-Richtung sicherstellen*“ dekomponiert.

Die Beziehungen zwischen Anforderungen und Anwendungsfällen werden, wie bereits dargelegt, durch `refine`-Beziehungen realisiert. Entsprechende Zellen der Matrix werden durch „*Refine*“ markiert.

Die Matrix bietet verschiedene Ansatzpunkte zur Verbesserung des Anforderungsmodells. Wenig spezifische Anforderungen werden nach ihrer Zuordnung zu bestimmten Anwendungsfällen untersucht. Eine weitere Ausformulierung der Anwendungsfälle bringt die Möglichkeit die Anforderung besser zu erfassen mit sich. Im Beispiel ist zu erkennen, dass der in der Nutzungsphase angesiedelte Anwendungsfall „*Sortieren von Muffins*“ die Anforderung „*Arbeitsraum bereitstellen*“ verfeinert, so dass auf der feingranularen dritten Ebene Wertebereiche angegeben werden können.

Darüberhinaus hilft die Zuordnung der Anforderung zu einer Phase des Entwicklungsprozesses, ihre Dringlichkeit und Priorisierung zu beurteilen. Beispielsweise ist eine Anforderung, die aus der Entsorgung z.B. Werkzeugmaschinen entsteht auf Grund der dort sehr langen Lebenszeiten mit nur geringer Priorität zu erfüllen. Außerdem ist die Dringlichkeit, d.h. die Notwendigkeit für diese Anforderung verlässliche Informationen zu beschaffen, ebenfalls gering. Im Automobilbau dagegen spielen Anforderungen aus dieser Lebensphase auf Grund von freiwilligen Selbstverpflichtungen der Hersteller eine große Rolle.

In vielen Fällen bewirken unterschiedliche Anwendungsfälle eine Verfeinerung der Anforderung in unterschiedlicher Weise. Beispielsweise wird eine Anforderung bezüglich des Laderaums eines Pkw durch den Anwendungsfall „*Familienausflug*“ auf einen anderen Wert festgelegt als durch den Anwendungsfall „*Einkauf im Möbelhaus*“. Im ersten Fall müssen vier bis fünf Reisekoffer verstaut und vier bis fünf Personen platziert werden. Im zweiten Fall müssen sperrige Möbelstücke untergebracht werden, die nicht selten länger als zwei Metern sind. Anforderungsspreizungen (vgl. Abschnitt 2.3.1) und ihre Ursachen sind auf diese Weise deutlich erkennbar.

Durch Priorisierung der Anwendungsfälle werden ebenfalls die Anforderungen priorisierbar. Der aus der Anforderungsspreizung resultierende Wertebereich wird entweder auf einen Wert reduziert und damit aufgehoben (z.B. indem niedrig priorisierte Anforderungen gestrichen werden) oder der Wertebereich wird durch eine Modularisierungs- oder Flexibilisierungsstrategie auf einige bestimmte Werte festgelegt. Im genannten Beispiel kann eine Modularisierung durch die Entwicklung eines Transportmoduls realisiert werden, welches im Bedarfsfall den Laderaum erweitert (z.B. Rucksack, Jetbag). Eine Flexibilisierung kann durch

das Umklappen oder Entnehmen der Rücksitze realisiert werden.

Zusammen mit der zuvor vorgestellten Stakeholder-Anwendungsfälle-Matrix lassen sich für bestimmte Anforderungen Stakeholder identifizieren, die als Informationsquelle dienen. Beispielsweise ist eine Anforderung „*Beleuchtung des Arbeitsraums vorsehen*“ für die manuelle Zuführung eines Werkstücks in eine Maschine festgelegt worden. Durch Befragung des Maschinenbedieners wird erfasst, welche Bereiche des Arbeitsraums ausgeleuchtet werden müssen und welche Objekte möglicherweise Schattenwürfe oder störende Reflexionen hervorrufen.

4.2.4 Anforderungen-Komponenten-Matrix

In der in Abb. 4.19 abgebildeten Anforderungen-Komponenten-Matrix werden die Anforderungen wie oben beschrieben in den Kopfspalten abgelegt. In den Kopfzeilen werden die Komponenten des Systems dargestellt (vgl. Abschnitt 4.1.7). Die oberen zwei Zeilen bilden die Struktur nach Packages ab. In der dritten Zeile befinden sich die Komponenten, die im Modell als Blöcke notiert werden. Beispielsweise ist der `<<block>>` Motor als Komponente dem Package *Antrieb* und dieses dem Package *Roboter* zugeordnet. Dies ermöglicht eine flexible Gliederung der Komponenten nach logischen Gesichtspunkten.

In den Zellen der aufgespannten Matrix werden die Zuordnungen der Anforderungen zu Komponenten durch „*Allocate*“ markiert. Dabei handelt es sich nicht um 1:1-Zuordnungen. Viele Anforderungen werden erst durch das Zusammenspiel mehrerer Komponenten erfüllt und viele Komponenten tragen zur Erfüllung mehrerer Anforderungen gleichzeitig bei. Beispielsweise trägt das Gelenk eines Parallelroboters durch den überstreichbaren Gelenkwinkel zur Erfüllung einer großen Bewegungsweite bei. Ebenso hat die gewählte kinematische Länge der Glieder einen großen Einfluss auf die Bewegungsweite.

Die Matrix zeigt übersichtlich, welche Anforderungen noch keinen Komponenten zugeordnet sind. Falls Anforderungen einem Package zugeordnet wurden, wird überprüft, ob eine bestimmte Komponente dieses Packages als konkretere Zuordnung geeignet ist.

		Komponenten	Roboter				
				Antrieb		Mech. Komponenten	
					Motor		Gelenk
Anforderungen							
hohe Antriebsleistung			Allocate	Allocate	Allocate		
	Maximale Leistungsaufnahme		Allocate	Allocate	Allocate		
Arbeitsraum breitstellen			Allocate			Allocate	Allocate
	notwendige Bewegungsweite		Allocate			Allocate	Allocate
		... in x-Richtung	Allocate			Allocate	Allocate
		... in y-Richtung	Allocate			Allocate	Allocate
		... in z-Richtung	Allocate			Allocate	Allocate

Abbildung 4.19: *Ausschnitt aus einer Anforderungen-Komponenten-Matrix für Parallelroboter.*

Bei gespreizten Anforderungen wird überprüft, ob die Komponente flexibel oder modular aufgebaut werden kann ohne gleichzeitig eine Vielzahl weiterer Anforderungen zu beeinflussen. Für konfliktäre Anforderungen hinsichtlich der gleichen Komponente wird untersucht, ob zur Konfliktauflösung eine der Anforderungen auf eine andere Komponente verlagert oder die Komponente geeignet zerlegt werden kann.

Werden die Anforderungen hinsichtlich ihrer Priorisierung markiert, so ist diese Auswertematrix ein geeignetes Mittel, um den Wert einzelner Komponenten für das Gesamtsystem zu beurteilen. Erfüllt eine Komponente beispielsweise eine große Anzahl von Festforderungen, die nicht von Anforderungsspreizungen betroffen sind, so eignet sie sich besonders als Grundbaustein in einem Baukastensystem. Sind wenige Anforderungsspreizungen in einer prinzipiell als Grundbaustein geeigneten Komponente vorhanden, wird untersucht, wie (z.B. durch Differenzierung) die Anforderungsspreizungen von der Komponente gelöst werden können. Auf der anderen Seite werden Komponenten identifiziert, die im Wesentlichen nur Anforderungsspreizungen zugeordnet sind und sich daher für Module eignen, die für bestimmte Anwendungsfälle austauschbar sind.

Des weiteren werden Komponenten identifiziert, die hauptsächlich Wünsche erfüllen. Diese Komponenten werden bei der Entwicklung zurückgestellt oder als Zusatz für eine „gehobene Ausstattung“ vorgesehen.

Einen besonderen Stellenwert erhält diese Matrix, falls Änderungen berücksichtigt werden müssen. Änderungsbegehren resultieren häufig entweder aus geänderten Randbedingungen (Eingangsgrößen der Produktentwicklung, vgl. Abb. 1.1) oder aus der Untererfüllung einzelner Anforderungen (vgl. auch Abschnitt 2.3.2). Erstere sind kaum planbar und können am ehesten als „höhere Gewalt“ bezeichnet werden. Änderungsbegehren auf Grund von Anforderungsuntererfüllung entstehen zwangsläufig, da Produkteigenschaften nicht direkt vom Entwickler festgelegt werden können (z.B. Maximalgeschwindigkeit). Der Synthese folgt daher immer eine Analyse mit häufig umfangreichen Tests. Werden Anforderungen nicht erfüllt, so können in der Matrix leicht die Komponenten identifiziert werden, die angepasst werden müssen. Gleichzeitig zeigt sich, welche anderen—unter Umständen bereits erfüllten—Anforderungen ebenfalls durch eine Anpassung betroffen sind.

4.2.5 Testkriterien-Testfälle-Matrix

Die Auswertematrix in Abb. 4.20 stellt die Testkriterien den Testfällen gegenüber. Hierfür wurde das für die Anforderungen-Anwendungsfälle-Matrix programmierte Makro abgewandelt. Die Testkriterien basieren auf dem Stereotyp `<<requirement>>` und die Testfälle auf `<<UseCase>>` und wurden durch die neuen bzw. erweiterten Stereotypen `<<testcriterion>>` und `<<TestCase>>` erweitert beschrieben (vgl. Abschnitt 4.1.8). Ihre Position im Modell ist durch die Packagestruktur vorgegeben, d.h. testrelevante Objekte befinden sich nur im Package

		Testfälle	TC15	TC19	TC26
			Packageskizzen überprüfen	Kollisionskontrolle im CAD	Vereinfachte Lebensdauerbetr.
Testkriterien					
tc_Lebensdauer	tc_Gestelllebensdauer				Verify
	tc_Komponentenlebensdauer				Verify
tc_Zugänglichkeit zum Arbeitsraum			Verify	Verify	Verify
	tc_Materialfluss nicht durch Gestellbauraum verhindern		Verify	Verify	
	tc_Manipulation durch Wartungstechniker ermöglichen		Verify	Verify	

Abbildung 4.20: *Ausschnitt aus einer Testkriterien-Testfälle-Matrix für Parallelroboter.*

`Testen`. In den aufgespannten Zellen werden die Beziehungen durch „*verify*“ gekennzeichnet.

In der Matrix wird angegeben, welche Testkriterien durch welche Testfälle validiert werden. Im gezeigten Beispiel sind die Testfälle festgelegten Testfallklassen zugeordnet. Beispielsweise wird mit Hilfe von Packageskizzen überprüft, dass ein Wartungstechniker Manipulationen innerhalb des Arbeitsraums durchführen kann. Das selbe Testkriterium wird durch Kollisionskontrollen im CAD-System überprüft. Die beiden Testfälle sind verschiedenen Testfallklassen zugeordnet. Im ersten Fall (TC15) ist ein abstraktes und im zweiten (TC19) ein virtuelles Testobjekt notwendig, um den Testfall durchzuführen. Im zweiten Testfall ist zwar mit einer höheren Genauigkeit zu rechnen, allerdings ist der Aufwand größer und der mögliche Testzeitpunkt liegt später im Produktentwicklungsprozess.

In Abschnitt 4.1.8 wurde dargelegt, dass die Testfälle ein Merkmal enthalten, welches die Phase im Produktentwicklungsprozess angibt, in der der Test durchgeführt werden soll. Werden die Testfälle zeitlich von links nach rechts und gleichzeitig die Testkriterien nach Risiko von oben nach unten sortiert, dann sollten die Matrixeinträge im Wesentlichen von links oben nach rechts unten verteilt sein, d.h. eine Diagonal bilden. Dann ist sichergestellt, dass stark risikobehaftete Testkriterien möglichst früh getestet werden. Zumindest wird durch abstrakte Tests die richtige Richtung der angedachten Lösung sichergestellt. In geeigneten Abständen werden diese Testkriterien dann bei immer konkreteren Randbedingungen erneut getestet. Gleichzeitig sollten Testkriterien mit geringem Risiko erst spät getestet werden, um den Aufwand zu minimieren.

Zur Minimierung des Testaufwands sollen möglichst viele Testkriterien mit dem gleichen Testfall getestet werden. Die Matrix wird daher nach Testfällen mit wenig Abdeckung untersucht. Anschließend wird überprüft, ob ein ähnlicher Testfall der gleichen Testfallklasse so angepasst werden kann, dass damit zusätzlich die Überprüfung dieser Testkriterien ermöglicht wird. Ist das der Fall, kann der ursprüngliche Testfall vom Testplan gestrichen werden und der Aufwand verringert sich.

4.2.6 Anforderungen-Anforderungen-Matrix

In Abb. 4.21 ist ein Beispiel für eine Anforderungen-Anforderungen-Matrix gezeigt. Diese stellt alle Anforderungen in den Kopfspalten dar (vgl. Abschnitt 4.2.3 und Abschnitt 4.2.4) und wiederholt sie transponiert in den Kopfzeilen. Dadurch wird eine symmetrische Matrix erzeugt. In den aufgespannten Zellen werden die Beziehungen genauer charakterisiert. In dem Formular aus Abb. 4.16 kann angewählt werden, welche Beziehungen dargestellt werden sollen. Wird die `trace`-Beziehung gewählt, so werden in der Matrix entsprechende Zellen mit „*trace*“ markiert.

In Abschnitt 3.4.2 wurden die Merkmale von Beziehungen genauer beschrieben. Der hier verwendeten `trace`-Beziehung wurden für eine genauere Beschreibung und Auswertung folgende *Tag Definitions* zugefügt:

- `granularity`: tbd (to be defined), same level, different level
- `quantifiability`: tbd, qualitative, quantitative, quantifiable
- `directions`: tbd, unidirectional, mutual
- `intensity`: tbd
- `support`: tbd, positive, negative, exclusionary
- `linkage`: tbd, direct, indirect
- `instance`: tbd, conditional, local, temporal

Für jedes Merkmal wird bei Bedarf ein neues Tabellenblatt erzeugt und die jeweilige Ausprägung in den entsprechenden Zellen angegeben. Das Merkmal der gerichteten bzw. wechselseitigen Abhängigkeit wird dadurch ausgedrückt, dass die Matrix von Spalte zu Zeile gelesen wird. D.h. die Anforderung der Spalte

Trace-Matrix / Anforderungen	Anforderungen	Betriebskräfte und -mom	x-Rea	x-Rea	Gesamtgewicht	Gewic	Gewic	max.Ner	innerbetrieblichen Transp	Staple	ohne
Betriebskräfte und -momente		■			trace	trace	trace	trace			
	x-Reaktionskräfte		■		trace						
	x-Reaktionsmomente			■	trace						
Gesamtgewicht		trace	trace	trace	■			trace	trace	trace	trace
	... eines Gelenkes	trace				■					
	... eines Stabes	trace					■				
max.Nennlast		trace			trace			■			
innerbetrieblicher Transport					trace				■		
	Staplertransport				trace					■	
	ohne Hifsmittel				trace						■

Abbildung 4.21: *Ausschnitt aus einer Anforderungen-Anforderungen-Matrix für Parallelroboter.*

Tabelle 4.2: *Schematische Darstellung einer „Support"-Matrix zur Verdeutlichung der halbautomatischen Zielkonfliktanalyse.*

	A1	A2	A3	A4
A1		+	+	?
A2				+
A3				-
A4			-	

beeinflusst die Anforderung der entsprechenden Zeile. Falls eine wechselseitige Beziehung vorliegt, wiederholt sich der Eintrag achsensymmetrisch zur Hauptdiagonalen.

Die so erzeugte „*Support*"-Matrix bildet die Basis für eine weitere Analyse des Anforderungsmodells. Basierend auf den Arbeiten zu einem Anforderungstool am Institut für Konstruktionstechnik [Fra02c] [Fra05d] [Ste06b] wurde eine halbautomatische Zielkonfliktidentifikation realisiert. Wird im Formular der Abb. 4.16 die Zielkonfliktanalyse aktiviert, identifiziert ein Makro auf Basis der vorhandenen Beziehungen mögliche weitere Beziehungen. Beispielsweise steht bei konventionellen Gelenken von Parallelrobotern ein kleines Gelenkspiel mit geringer Gelenkreibung in Konflikt (z.B. [Pav06], [Ste07b]). Die geringe Reibung unterstützt unter anderem einen geringen Gelenkverschleiß, welcher längere Wartungs- und Austauschintervalle ermöglicht. Da ein kleines Gelenkspiel die Genauigkeit des Roboters unterstützt, folgt hier ebenfalls ein Konflikt zwischen langen Wartungsintervallen und hoher geforderter Genauigkeit.

Tabelle 4.2 veranschaulicht das Vorgehen an einem einfachen, abstrakten Beispiel: Die Anforderung $A1$ unterstützt $A2$ und $A3$ unidirektional. $A2$ unterstützt $A4$ unidirektional. $A3$ und $A4$ stehen in einem wechselseitigen Konflikt. Eine Verfolgung der Anforderungsbeziehungen über zwei Anforderungen ausgehend von $A1$ ergibt folgende Wege:

$$A1 \xrightarrow{+} A2 \xrightarrow{+} A4 \tag{4.2}$$

$$A1 \xrightarrow{+} A3 \xrightarrow{-} A4 \tag{4.3}$$

Daraus ergibt sich eine formal mögliche Beziehung in der mit „?" gekennzeichneten Zelle von Tabelle 4.2. Die Auswertung der Gl. (4.2) und Gl. (4.3) ergibt folgenden Zusammenhang:

$$A1 \xrightarrow{+} A4 \tag{4.4}$$

$$A1 \xrightarrow{-} A4 \tag{4.5}$$

Das bedeutet, der erste Weg lässt auf eine Unterstützung der zweite Weg auf einen zusätzlichen Konflikt schließen. Sobald eine konfliktäre Beziehung in dem Weg enthalten ist, besteht die Wahrscheinlichkeit, dass auch die beiden Enden

in einem Konflikt zueinander stehen ($+ \wedge - \simeq -$). Sind alle Beziehungen eines Weges unterstützend, so ist wahrscheinlich eine Beziehung zwischen den Enden ebenfalls unterstützend ($+ \wedge + \simeq +$). Andererseits ist es zwar möglich aber unwahrscheinlich, dass bei zwei konfliktären Beziehungen zwischen den Enden eine Unterstützung vorliegt ($- \wedge - \not\simeq +$). Da Zielkonflikte häufig unterschiedliche Ursachen haben, kommt es nur in Einzelfällen vor, dass sich zwei Konflikte so auslöschen, das die Enden in einer Unterstützung zueinander stehen.

Die Deutung, ob hier eine Unterstützung, ein Konflikt oder überhaupt keine Beziehung vorliegt, kann aus den dargelegten Gründen nicht automatisch erfolgen. Ein Entwickler muss seine Intelligenz einsetzen, um die Beziehung zu analysieren und eine Entscheidung zu fällen. Als erstes Hilfsmittel werden die in Gl. (4.2) und Gl. (4.3) angegebenen Wege in einem Kommentar der entsprechenden Zelle angegebenen. So bleiben eventuell vorhandene Einträge in den Zellen aus bekannten direkten Beziehungen erhalten und im Kommentar werden mehrere unterschiedliche indirekte Wege dargestellt. Im Einzelfall muss entschieden werden, welche Beziehung als dominant betrachtet wird.

Zur Beurteilung der Beziehungen werden die oben beschriebenen Merkmale der involvierten Beziehungen berücksichtigt. Ist z.B. die Stärke (`intensity`) eines Weges deutlich geringer als die eines zweiten Weges, so ist letzterer mit hoher Wahrscheinlichkeit der dominante. Bei quantifizierten Beziehungen (`quantifiable`) wird der Einfluss konkret gefasst (vgl. Abschnitt 4.1.6). Ist eine Beziehung als quantifizierbar gekennzeichnet, so ist der Aufwand gerechtfertigt hier eine quantitative Beschreibung zu erarbeiten, um den Einfluss genau zu untersuchen (vgl. Parametrikdiagramm Abb. 4.10). Zeigt das Merkmal `granularity` eine Beziehung zwischen unterschiedlichen Ebenen an, sollte zunächst untersucht werden, ob die Beziehung nicht auf der gleichen Ebene zu beschreiben ist, damit eine sichere Aussage über ihre Art möglich ist.

Die Merkmale `directions` und `instance` geben Hinweise für den Umgang mit den identifizierten Zielkonflikten. Beispielsweise sind unidirektionale Beziehungen hinsichtlich ihrer Eignung als Verhinderer von Änderungsauswirkungen (*propagation blocker*) zu untersuchen. Der Umstand (`instance`) hilft, sich für eine Zielkonfliktlösungsstrategie zu entscheiden (vgl. Abschnitt 3.4.2). Bei einer konditionalen Beziehung lässt sich die Ursache des Zielkonflikts finden und beheben, beispielsweise indem ein anderes Fertigungsverfahren gewählt wird. Lokale Beziehungen lassen sich z.B. durch örtliche Verlagerung oder Entkopplung der Beziehungspartner entschärfen. Temporale Beziehungen lassen sich durch gezielte Flexibilität bzw. Adaptivität auflösen.

Die Identifikation neuer Beziehungen (insbesondere Zielkonflikte) kann nur halbautomatisch erfolgen. Es stellt sich die Frage: Warum dann nicht gleich die gesamte Matrix durch einen erfahrenen Entwickler überprüfen lassen?

Zur Beantwortung dieser Frage soll eine kurze Rechnung herangezogen werden: Ein Projekt mittlerer Größe erhält schnell ein Anforderungskollektiv mit bis zu 500 Anforderungsobjekten. Daraus ergeben sich 249.500 zu überprüfende

Zellen. Selbst bei Betrachtung einer Halbmatrix bleiben 124.750 Zellen. Falls ein Entwickler eine Minute Zeit zum Beurteilen der Anforderungsbeziehung benötigt, ist er über ein Arbeitsjahr damit beschäftigt die Matrix auszufüllen. Diese Zeit ist in der Industrie nicht vorhanden. Außerdem muss diese stupide Arbeit als stark fehlerbehaftet angesehen werden. Dadurch ist die Aussagekraft der Matrix als gering einzuschätzen.

Bei der vorgestellten Methode werden die Beziehungen bereits während der Modellierung gesetzt. Durch die übersichtliche grafische Darstellung wird eine höhere Qualität der Ergebnisse sichergestellt (vgl. Abschnitt 2.2.5). Anschließend werden ausschließlich die Beziehungsmöglichkeiten untersucht, die eine hohe Wahrscheinlichkeit einer Beziehung mit sich bringen. Dadurch ist keine—ohnehin kaum nachweisbare—vollständige Erfassung aller Beziehungen möglich. Das Verhältnis zwischen Nutzen und Aufwand ist allerdings deutlich größer.

4.2.7 Zielverfolgung

Jeder Produktentwicklung soll eine bestimmte Zielsetzung vorstehen (vgl. Abschnitt 2.1.1 und Abschnitt 2.1.3). Im Projektverlauf werden ständig neue Informationen gewonnen und in Anforderungen beschrieben. Es werden Konzepte entwickelt und teilweise wieder verworfen. Die Randbedingungen (z.B. strategische Ausrichtung der Unternehmen, Verwerfungen am Markt) ändern sich ständig. Einige Informationen (und Anforderungen) werden dadurch überflüssig oder unterstützen nicht (oder nicht mehr) die Ziele der Entwicklung.

In der zur Modellierung verwendeten, kommerziellen Software ist bereits die Möglichkeit enthalten `satisfy`-Matrizen in Microsoft Excel erstellen zu lassen. Diese Matrix wird für die Auswertung verwendet, daher musste kein eigenes Makro zu diesen Zweck programmiert werden.

Wurde das Modell wie beschrieben aufgebaut, entstehen Matrizen, die die Erfüllungsbeziehungen der Anforderungen zu den Zielen und der Ziele zu den Objekten der Systemidee abbilden. Anhand dieser Matrizen ist erkennbar, ob es Anforderungen gibt, die nicht zu einer Zielerfüllung beitragen. Diese stammen z.B. als Restriktionen aus der Produktumgebung oder sind z.B. als funktionale Anforderungen notwendig, um einen bestimmten Anwendungsfall zu erfüllen. Es ist kritisch zu hinterfragen, ob diese Anforderungen zum gegenwärtigen Projektstand noch gerechtfertigt sind. Beispielsweise falls bestimmte Anwendungsfälle aus der Systemgrenze genommen oder bestimmte Ziele verändert wurden. Beispielsweise ist die „*Reinraumeignung*“ nicht mehr zu erfüllen, falls die Anwendungsfälle der Mikrochipmontage nicht mehr betrachtet werden. Sollen aus unternehmensstrategischen Gründen keine rekonfigurierbaren Systeme mehr angeboten werden, fallen auch dadurch eine Reihe von Anforderungen (z.B. „*Lösbare Verbindungen im Gestell verwenden*“) weg.

Für Anforderungen und Komponenten werden im Bedarfsfall Diagramme erzeugt, die den Weg zum Ziel darstellen. Dazu werden vom Ausgangselement aus-

gehend sukzessive durch die `populate`-Funktion die Erfüllungsbeziehungen nachverfolgt. Es entsteht ein Netz, dass die verschiedenen Wege darstellt auf denen das Objekt zur Zielerfüllung beiträgt (vgl. Abb. 4.9). Dadurch kann die Wichtigkeit bzw. der Wert einer Anforderung für das Gesamtprodukt visualisiert und daher besser eingeschätzt werden.

4.2.8 Projektüberwachung

Die Überwachung des Projektes erfolgt durch Kennzahlen, die in regelmäßigen Abständen erfasst werden. Grundlage hierfür sind die Modelldaten und die zuvor beschriebenen Matrizen. Die Daten werden in Microsoft Excel erfasst, weiterverarbeitet (z.B. Kennzahlen gebildet) und grafisch aufbereitet.

Abb. 4.22 zeigt ein Beispiel zur Verdeutlichung des zeitlichen Verlaufs der Anforderungserfassung, wie er für viele Projekte zu erwarten ist. Die Gesamtanzahl an Anforderungen steigt zunächst stark an und flacht im weiteren Verlauf ab. Falls sich bereits in frühen Entwicklungsphasen eine Stagnation einstellt, deutet dies auf eine Vernachlässigung der Anforderungserfassung hin. Es kann zu keinem Zeitpunkt während der Entwicklung beurteilt werden, wieviele Anforderungen für ein vollständiges Anforderungsmodell notwendig sind, zumal die Vollständigkeit mit dem möglichst minimalen Umfang konkurriert. Es kann höchstens im Nachhinein festgestellt werden, dass eine wichtige Anforderung nicht berücksichtigt wurde. daher ist es sinnvoller von einem für einen bestimmten Entwicklungsschritt hinreichenden Anforderungsmodell zu sprechen. Die Dokumentation der zeitlichen Entwicklung gibt einem erfahrenen Projektleiter ein Instrument zur Beurteilung der Qualität der Anforderungsklärung an die Hand. Er kann damit z.B. beurteilen, ob zu einem bestimmten Zeitpunkt ein hinreichendes Anforderungsmodell vorliegt. Der Effekt der frühen Stagnation der Anforderungsklärung wurde z.B. von *Jung* [Jun06] beschrieben (vgl. Abb. 4.23). Nach anfänglich intensiver Aufgabenklärung sinkt demnach die Intensität sehr stark ab, neue Anforderun-

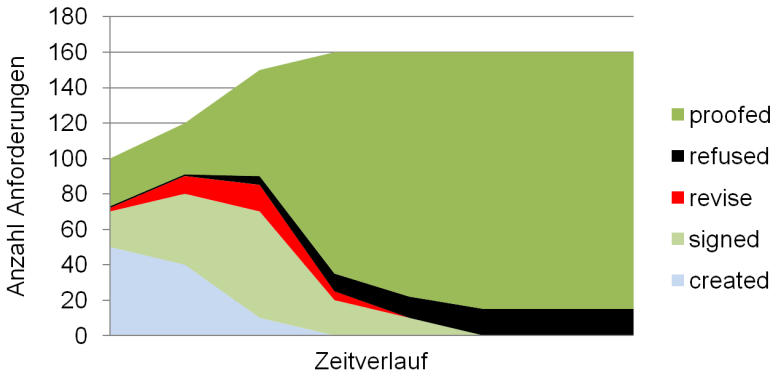


Abbildung 4.22: *Beispiel zur Verdeutlichung des zeitlichen Verlaufs der Anforderungserfassung mit Status Kennzeichnung.*

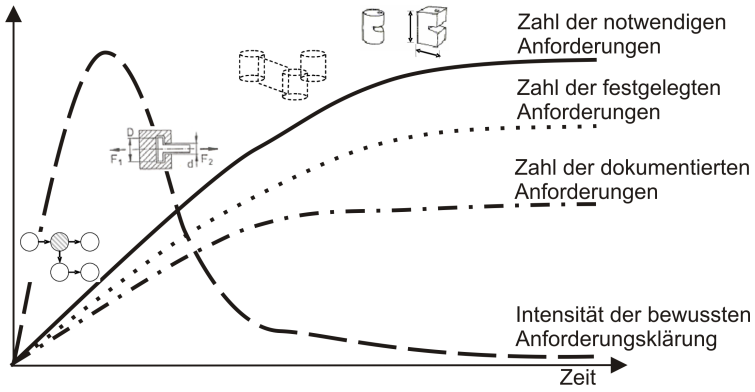


Abbildung 4.23: *Zeitlicher Verlauf der Anforderungsklärung, nach [Jun06, S. 18].*

gen werden kaum noch spezifiziert und selten dokumentiert.

Die Anzahl von Anforderungen allein ist für eine Beurteilung des Projektfortschritts nicht ausreichend. Daher wird im entwickelten Ansatz jeder Anforderung ein Status zugewiesen (vgl. Abschnitt 4.1.6). Der Status der Anforderungen ändert sich zu Beginn schnell, da neu erfasste Anforderungen nach und nach freigegeben werden. Die mit `created` und `signed` markierten Anforderungen werden im Zeitverlauf verschwinden. Bei normalem Verlauf der Entwicklung wird der Anteil der als zu überarbeiten (`revise`) markierten Anforderungen zuerst ansteigen und später ebenfalls verschwinden.

Der Anteil an gelöschten Anforderungen (`refused`) steigt im Zeitverlauf an. Nehmen gelöschte Anforderungen einen hohen prozentualen Wert an, so besteht Optimierungsbedarf bei der Anforderungserfassung: Es werden anfänglich zu viele unbrauchbare oder redundante Anforderungen erfasst. Für eine genauere Untersuchung der Ursachen, die zum Löschen geführt haben, werden die beim Löschen gesetzten Kommentare (`rationale`) berücksichtigt.

Weiterhin ist es sinnvoll, die Priorisierung zu beobachten (Abb. 4.24 a). Hier wird die prozentuale Aufteilung in Fest-, Mindest- und Wunschforderungen do-

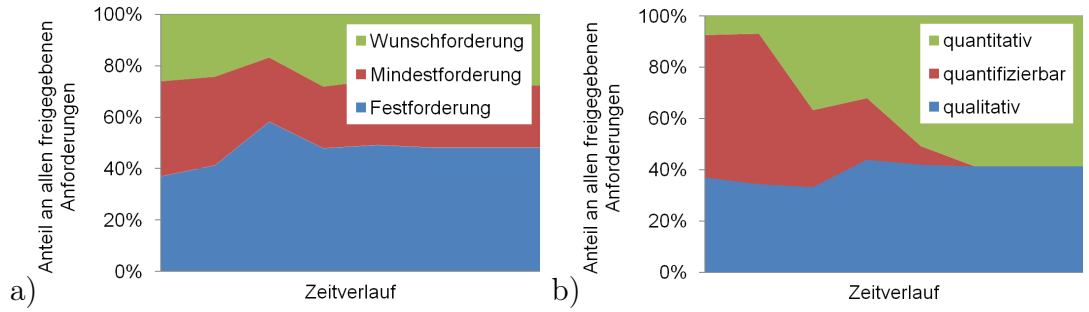


Abbildung 4.24: *Aufteilung der Priorisierung (a) und Quantifizierung (b) von Anforderungen im Zeitverlauf.*

kumentiert. Ein zu großer Anteil an Wünschen kann darauf hinweisen, dass die Priorisierung nicht sinnvoll durchgeführt wurde. Insbesondere in frühen Entwicklungsphasen sollte der Schwerpunkt der erhobenen Anforderungen auf Fest- und Mindestforderungen liegen [Dre93, S. 120f].

Betrachtet man das Merkmal der Quantifizierung (Abb. 4.24 b), sollte die Ausprägung „quantifizierbar“ im Zeitverlauf vollständig verschwinden. Der Großteil der Anforderungen sollte quantitativ vorliegen, so dass er eindeutig bewertbar ist.

Abb. 4.25 a) zeigt ein Diagramm, dass die Validierbarkeit und die Erfüllbarkeit der freigegebenen Anforderungen prozentual im Zeitverlauf darstellt. Dabei wird eine Anforderung formal als validierbar bezeichnet, falls sie mit mindestens einem Testkriterium in einer `satisfy`-Beziehung steht. Dadurch soll verhindert werden, dass nicht überprüfbare Anforderungen das Anforderungskollektiv unnötig aufblähen (vgl. Abschnitt 4.1.8).

Eine Anforderung wird formal als erfüllbar bezeichnet, falls sie mit mindestens einer Komponente in einer `allocate`-Beziehung steht. Dadurch wird gewährleistet, dass keine Anforderungen bei der Umsetzung „vergessen“ werden.

In Abb. 4.25 b) ist dargestellt, wie hoch der Anteil an sicheren bzw. unsicheren Anforderungen im Anforderungskollektiv ist. Dazu wird das Merkmal `certainty` (vgl. Abschnitt 4.1.6) der Anforderungen genutzt. Je größer der Anteil an unsicheren Anforderungen ist, desto höher ist die Wahrscheinlichkeit von späteren Änderungen und desto höher ist das Entwicklungsrisiko.

Abb. 4.26 zeigt ein Diagramm zur Betrachtung der zeitlichen Entwicklung der Beziehungsarten. Es wird die prozentuale Verteilung der Merkmale „*Ausschluss*“, „*Konflikt*“ und „*Unterstützung*“ auf alle gesetzten `trace`-Beziehungen dargestellt. Je größer der Anteil an ausschließenden und konfliktären Beziehungen ist, desto aufwändiger wird eine ausgeglichene Erfüllung der Anforderungen durch das Produkt.

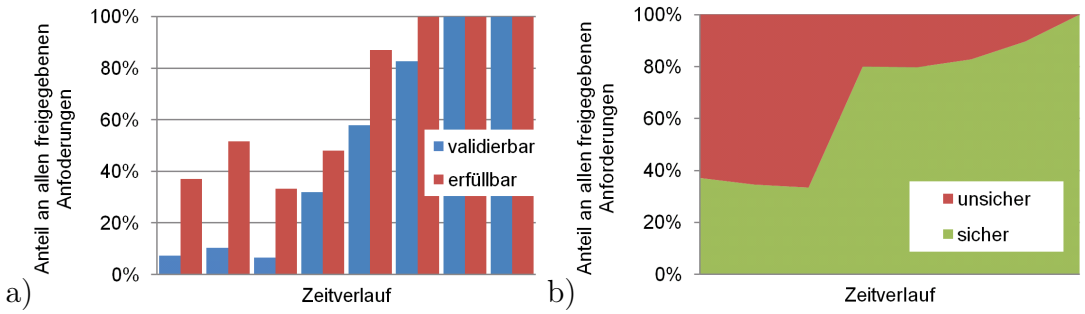


Abbildung 4.25: *Anteil a) verifizier- bzw. erfüllbarer und b) sicherer Anforderungen der freigegebenen Anforderungen im Zeitverlauf.*

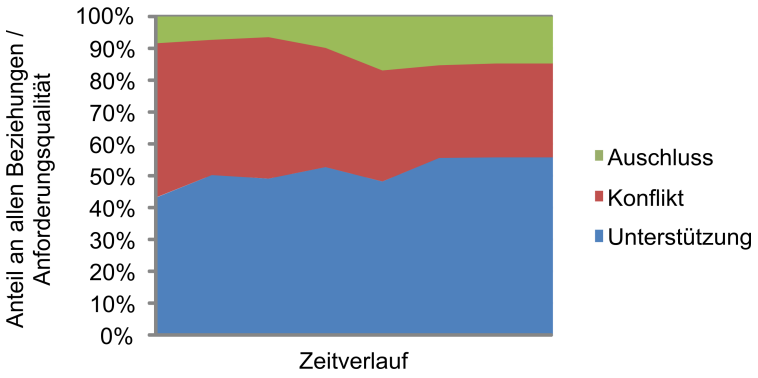


Abbildung 4.26: *Arten von Beziehungen zwischen Anforderungen im Zeitverlauf.*

4.3 Vorgehen zur Baukastenentwicklung

In Abschnitt 2.3 wurde bereits darauf hingewiesen, dass eine Vielzahl von Vorgehensweisen zur Entwicklung von Baukastensystemen existiert. Diese Vorgehen haben Stärken und Schwächen in unterschiedlichen Bereichen. Hier soll demonstriert werden, wie die vorgestellte Anforderungsmodellierung für eine sinnvolle Baukastenentwicklung genutzt werden kann. Im Allgemeinen ist eine Kombination mit weiteren Methoden sinnvoll.

Neben dem üblichen Verständnis von Baukastensystemen wird hier insbesondere auch die Möglichkeit der Rekonfiguration betrachtet. Das Baukastensystem hat in diesem Fall den Charakter eines Anwenderbaukastens bei dauernder „Nutzungsdauer“ (vgl. Tabelle 2.1), wobei die Rekonfiguration „online“ im Betrieb oder „offline“ stattfinden kann. Bei letzterem wird der Betrieb des Systems unterbrochen und nach erfolgter Rekonfiguration wird es neu in Betrieb genommen.

Das entwickelte Vorgehen orientiert sich an der Darstellung in Abb. 4.1. Die Bausteine der Produktentwicklung werden bearbeitet, durch die angegebenen Partialmodelle abgebildet und durch die Auswertemechanismen unterstützt. Falls Vorgängerprodukte vorhanden sind, wird auf ein existierendes Modell zurückgegriffen. In diesem Fall ist dann bereits umfangreiches Wissen (z.B. Anforderungen, Beziehungen) hinterlegt und das Modell muss „nur“ an die Baukastenproblematik angepasst und erweitert werden. Falls kein Vorgänger existiert—es sich um eine vollständige Neuentwicklung handelt—werden alle Partialmodelle sukzessive, iterativ erarbeitet und erweitert. Hier werden nur die Aktivitäten aufgeführt, die sich von einer „normalen“ Entwicklung unterscheiden bzw. Besonderheiten aufweisen.

In Abb. 4.27 ist das Vorgehen zur interdisziplinären, anforderungsgesteuerten Baukastenentwicklung in Form eines Ablaufplans dargestellt. Es besteht aus fünf Schritten, die im Folgenden näher beschrieben werden:

1. Anwendungsfälle strukturieren
   Das Strukturieren der relevanten Anwendungsfälle im Produktlebenslauf-

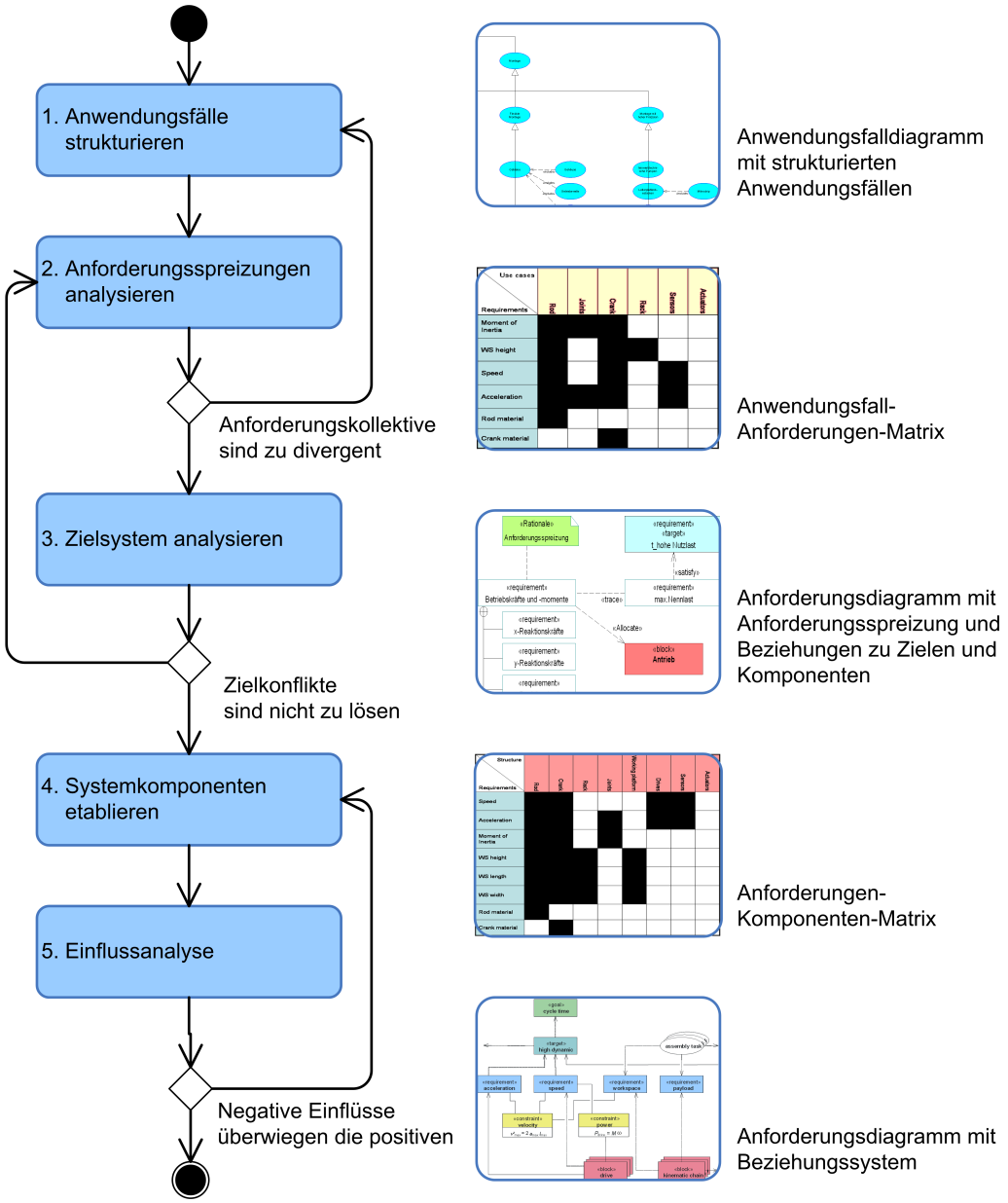


Abbildung 4.27: *Ablaufplan für eine interdisziplinäre, anforderungsgesteuerte Baukastenentwicklung und die für die einzelnen Schritte zur Verfügung stehenden Hilfsmittel.*

modell (s. Abschnitt 4.1.3) z.B. nach unterschiedlichen Aufgaben (Handhabung oder Montage bei Robotersystemen) oder Einsatzgebieten (Befahren eines Ozeans oder eines Flusses bei Booten) bildet den ersten Schritt des vorgestellten Vorgehens. Anschließend werden `refine`-Beziehungen zu bereits erfassten Anforderungen festgelegt bzw. die verbundenen Anforderungen werden im Anforderungsmodell (s. Abschnitt 4.1.6) genauer beschrie-

ben. Hierbei werden Anforderungen in der Weise dekomponiert, dass es für eine übergeordnete Anforderung Kinderobjekte gibt, die sich lediglich in der Ausprägung ihres Wertebereichs unterscheiden. Es wird also eine Anforderungsspreizung (vgl. Abschnitt 2.3.1) für bestimmte Anforderungen modelliert.

2. Anforderungsspreizungen analysieren
   Die in Abschnitt 4.2.3 beschriebene Anforderungen-Anwendungsfälle-Matrix hilft zu erkennen in welchen Phasen des Produktlebens die Anforderungsspreizungen auftreten. Handelt es sich beispielsweise um Anwendungsfälle während des Betriebes, die zudem häufig wechseln, ist entweder eine Anpassung der Produkteigenschaften durch adaptive Elemente oder eine dynamische Rekonfiguration des Systems zu empfehlen. Ergibt die Analyse, dass die die Anforderungsspreizung verursachenden Anwendungsfälle niemals im Lebenslauf desselben Produktes auftreten, sollte ein Herstellerbaukasten realisiert werden. Bereits in diesem Schritt lassen sich Anforderungskollektive für bestimmte Szenarien zusammenstellen, d.h. es gibt bestimmte Wertebereiche von verschiedenen Anforderungen, die niemals gleichzeitig erfüllt werden müssen (vgl. [Jes96, S. 4-12]). Beispielsweise ist der Wert für die erreichbare Höchstgeschwindigkeit eines Supersportwagens sehr hoch, während die Anzahl vorhandener Sitzplätze klein ist. Eine Variante eines Autos, die bei 8 Sitzplätzen über 300 km/h schnell fährt, wird nicht gefordert und muss in einem Baukastensystem nicht berücksichtigt werden.
   Im Anschluss an diesen Schritt erfolgt eine Beurteilung der Zweckmäßigkeit der verschiedenen Anforderungskollektive. Werden die Anforderungskollektive als zu divergent erachtet als das sie in einem Baukastensystem zu realisieren sind, müssen die Anwendungsfälle neu strukturiert werden.

3. Zielsystem analysieren
   Ausgehend von den als Spreizung identifizierten Anforderungen werden die übergeordneten Ziele untersucht, d.h. die `satisfy`-Beziehungen verfolgt (s. Abschnitt 4.1.2). Dazu wird eine `satisfy`-Matrix aufgebaut, die angibt welche Ziele durch die Anforderungsspreizung betroffen sind (vgl. Abschnitt 4.2.7). Gleichzeitig wird mit Hilfe der Matrix untersucht, welche Anforderungen mit den Zielen der Baukastenstrategie (vgl. „Stoßrichtungen“ in Abb. 2.14) zusammenhängen. Daraus ergibt sich ein Kollektiv von essentiellen Anforderungen für die Konzipierung des Baukastensystems sowie für die Gestaltung von Modulen und Schnittstellen. Diese können als die wesentlichen „Variantentreiber“ angesehen werden.
   Dieses Kollektiv wird nach Zielkonflikten untersucht. Dazu wird in der Anforderungen-Anforderungen-Matrix (vgl. Abschnitt 4.2.6) überprüft, ob für die relevanten Anforderungen eine `trace`-Beziehung gesetzt und als konfliktär markiert wurde. Im Folgenden wird untersucht, ob diese Zielkonflikte

im Baukastensystem z.B. durch eine geeignete Modulaufteilung oder eine der in Abschnitt 3.4.2 beschriebenen Konfliktlösungsstrategien aufgelöst werden können. Ist das nicht der Fall müssen die Anforderungsspreizungen erneut überprüft und dann ggf. die relevanten Anwendungsfälle neu festgelegt werden.

4. Systemkomponenten etablieren
   Es werden Beziehungen von den identifizierten Anforderungen zu Systemkomponenten und deren Parametern gesetzt bzw. analysiert. Hierzu dient die in Abschnitt 4.2.4 beschriebene Anforderungen-Komponenten-Matrix. Es wird untersucht, ob eine Komponente in der Lage ist die unterschiedlichen Anforderungen zu erfüllen oder—bei einem modularen Ansatz—welche Parameterstufungen für die Varianten des Moduls sinnvoll sind. Außerdem wird überprüft welche Modulaufteilung sinnvoll erscheint. Muss eine Anforderung bzw. ein bestimmter Wertebereich laut der Anwendungsfall-Analyse nur selten erfüllt werden, so ist hier z.B. ein Kann-Baustein (vgl. [Pah07, S. 664] vorzusehen, der nur optional zu verwenden ist.

5. Einflussanalyse
   Die Parameter, die in einem Baukastensystem in unterschiedlichen Werten zu realisieren bzw. bei einer Rekonfiguration zu verändern sind, wurden identifiziert. Des weiteren wurde die Modulaufteilung festgelegt. Jetzt muss der Einfluss auf das Gesamtsystem untersucht werden. Dazu wird erneut die Anforderungen-Komponenten-Matrix hinzugezogen (Abschnitt 4.2.4). Für jeden anzupassenden Parameter wird der Einfluss auf das Gesamtsystem abgeschätzt. Da sich hier der Einfluss über mehrere Partialmodelle erstreckt, ist es zudem sinnvoll Diagramme zu erzeugen, die ausgehend von dem zu untersuchenden Parameter alle Beziehungen verfolgen lassen (vgl. Abschnitt 4.2.1). Dadurch wird ausgeschlossen, dass die Änderung eines Parameters außer den gewünschten positiven Effekten noch schwerwiegende negative Effekte auf das Gesamtsystem ausübt. Insbesondere müssen die Kostenziele einzuhalten sein. Stellt sich heraus, dass die negativen Einflüsse die positiven Einflüsse überwiegen, muss die Modulaufteilung bzw. die Wahl der Parameter verbessert werden. Gelingt das nicht, so muss das Zielsystem überdacht werden. Unter Umständen müssen sogar die Anwendungsfälle neu strukturiert und der Relevanzbereich neu festgelegt werden.

An den verschiedenen Entscheidungspunkten in diesem Vorgehen, werden durch *versioning* (vgl. Abschnitt 3.3.2) jeweils neue Varianten des Modells erzeugt, die verschiedene Möglichkeiten abbilden. Diese Varianten lassen sich miteinander vergleichen. Nicht zielführende Varianten werden in einer „Sackgasse“ beendet und es wird mit dem letzten sinnvollen Vorgänger oder einer alternativen Version weitergearbeitet.

Kapitel 5

Erprobung des entwickelten Konzepts

Das in Abschnitt 4 entwickelte Modellierungskonzept wird im Folgendem anhand einiger Beispiele erprobt. Zunächst soll an einer einfachen Aufgabe mit Studierenden untersucht werden, ob das Konzept eine generelle Akzeptanz für die praktische Nutzung erwarten lässt. Anschließend wird an einem komplexeren Beispiel aus dem Bereich der Robotik die Leistungsfähigkeit des Konzepts unter Beweis gestellt. Dazu wird zunächst das Robotermodell systematisch aufgebaut und dabei unter anderem kurz auf die Punkte Baukastenentwicklung (Abschnitt 5.2.7) und Identifikation von Testfällen (Abschnitt 5.2.8) eingegangen. Anschließend werden unter Zuhilfenahme des Modells Parameter für die Rekonfiguration des Robotersystems identifiziert. Eine weitere Anwendung findet sich in [Rie10]. Dort wird im Rahmen einer Studienarbeit mit Hilfe des in Abschnitt 4 beschriebenen Vorgehens ein modulares Bootskonzept erarbeitet.

5.1 Akzeptanzuntersuchung

Methoden, Methodiken oder Werkzeuge können nur dann als gut bezeichnet werden, falls sie von den Nutzern akzeptiert werden. Aus diesem Grund wird die Akzeptanz der entwickelten Anforderungsmodellierung im Rahmen einer Vorlesung mit Studierenden (den möglichen Nutzern von morgen) überprüft. Das Projekt ist hinsichtlich seiner Komplexität hinreichend einfach, um im Rahmen einer Vorlesung von Studierenden bearbeitet werden zu können.

Die Akzeptanz unterliegt nach *Jänsch* vielen Einflussfaktoren [Jän07, S. 47ff]. So lassen sich im Wesentlichen Darstellungs-, Lehr- und Anwendungsprobleme unterscheiden. Die im Rahmen dieser Arbeit durchgeführte einfache Untersuchung soll einen ersten generellen Überblick verschaffen und verzichtet daher auf eine tiefergehende Betrachtung der einzelnen Aspekte. Es soll hier zunächst lediglich untersucht werden, ob eine Anwendung der Methode und der Modellierung in der Praxis zu erwarten ist. Zur Verallgemeinerung der Ergebnisse und Feststellung der sinnvoll unterstützbaren Projekte (z.B. hinsichtlich Aufgabenkomplexität, Projektteamgröße) müssen weitere, detailliertere Untersuchungen folgen.

5.1.1 Aufgabe

Ein handelsübliches ferngesteuertes Modellauto ist mit einem Einzylinder-Zweitaktmotor (0,88 kW bei 19000 min^{-1}) ausgestattet und erreicht eine Spitzengeschwindigkeit von ca. 45 km/h. Es soll ein Derivat entwickelt werden, dass sich durch besseres Abgasverhalten auszeichnet, gleichzeitig aber eine akzeptable Fahrdynamik bereitstellen soll. Dazu wird der Antriebsstrang inkl. Motor durch eine Neukonstruktion ersetzt. Als Austauschmotor dient ein verbrauchsärmerer Viertakt-Motor, der allerdings eine wesentlich geringere Leistung bei gleichzeitig geringeren Drehzahlen aufweist ($0,29$ kW bei 9500 min^{-1}).

Zur Bewältigung der Aufgabe werden alle Phasen der Produktentwicklung durchlaufen. Im ersten Schritt wird die Aufgabe geklärt und Ziele und Anforderungen definiert. Anschließend wird mit Hilfe mathematischer Modelle im Computer Algebra System (CAS) Maple [Map10] eine Übersetzungsstrategie festgelegt. Hier werden die Motorcharakteristiken und Fahrzeugwiderstände berücksichtigt. Mit einem Programm zur Berechnung von Maschinenelementen (eAssistant [GWJ10]) werden schließlich die Zahnräder ausgelegt und zur abschließenden Gestaltung in das CAD-System Solid Edge [Sie10] exportiert. Die fertig gestalteten Bauteile werden als funktionsfähige ABS-Prototypen im Extrusionsverfahren hergestellt. Sie werden in das Derivat montiert und anschließend auf einer Rennstrecke getestet.

Abb. 5.1 ist ein Foto des getesteten Modellautos und der Komponenten. Vor dem Auto liegen links die ABS-Prototypen eines zweistufigen Getriebes. Deutlich zu erkennen sind die Brüche des kleineren Zahnrades auf Höhe der Bohrungen, die zur Befestigung auf der Schaltkupplung notwendig waren. Rechts daneben befinden sich die konventionell in Stahl gefertigten Prototypen eines einstufigen Getriebes mit deutlich höherer Untersetzung.

5.1.2 Versuchsgruppe

Die Teilnehmer sind Studierende des Diplomstudiengangs Maschinenbaus an der TU Braunschweig im fünften und siebten Semester. Sie besitzen ein konstruktionsmethodisches Hintergrundwissen. Die gebräuchlichen Vorgehensweisen und Hilfsmittel zur Anforderungsklärung und Dokumentation (z.B. Checklisten, Suchwürfel, Anforderungsliste) sind ihnen bekannt und wurden im Verlauf des Studiums in verschiedenen Projektarbeiten angewendet. Anwenderkenntnisse über die verwendete Notation SysML oder zur UML waren bei keinem der Teilnehmer vorhanden. Nach eigenen Einschätzungen verfügten alle Teilnehmer über Grundkenntnisse der Programmierung. Etwa zwei Drittel der Teilnehmer hatte geringe und ein Drittel bisher keine Erfahrungen mit Objektorientierung.

Die Aufgabe wurde in Teams zu drei bis vier Personen bearbeitet. Die Teamfindung und Arbeitsteilung erfolgte selbstständig durch die Teilnehmer.

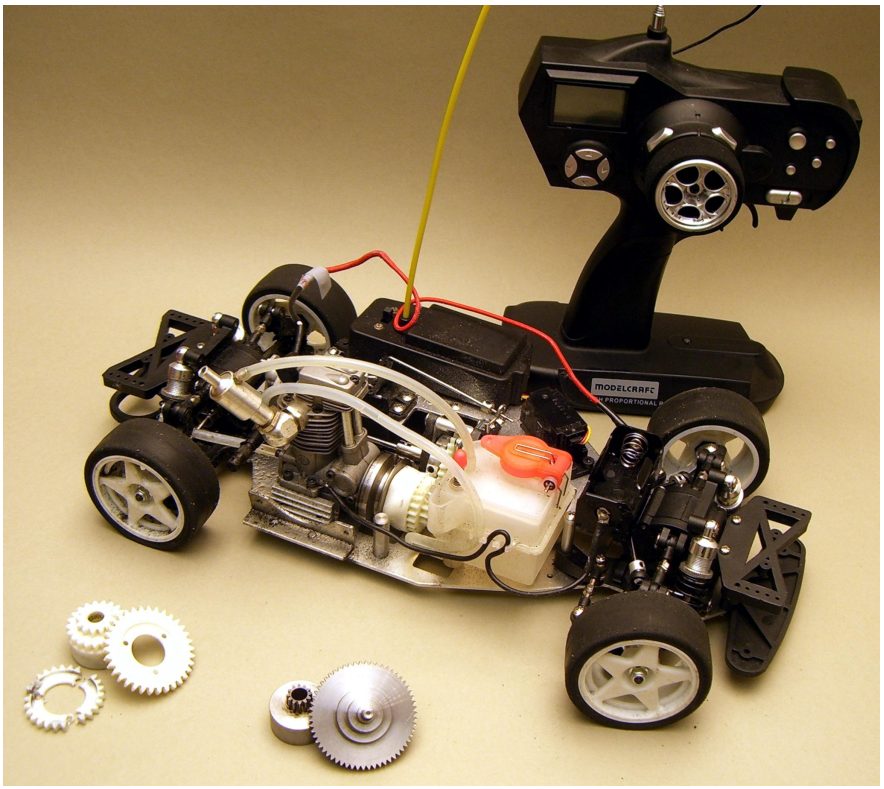


Abbildung 5.1: *Foto des Modellautos mit den ABS-Prototypen eines zweistufigen (links) und den konventionell gefertigten Stahl-Prototypen eines einstufigen (rechts) Getriebes.*

5.1.3 Versuchsdurchführung

Der Versuch wurde in zwei Stufen durchgeführt. Die erste Stufe fand im Wintersemester 2007/2008 mit 15 Probanden statt. Hier war es zunächst das Ziel die allgemeine Akzeptanz der Anforderungsmodellierung zu überprüfen. Außerdem sollte festgestellt werden, ob die Modellierung ohne aufwändige Einweisung bereits gute Ergebnisse liefern kann. Deshalb wurde lediglich eine kurze Einweisung (ca. 30 min) in die Notation und die Bedienung der Software hinsichtlich der reinen SysML-Anforderungsmodellierung gegeben.

Die zweite Stufe fand im darauf folgenden Jahr im Wintersemester 2008/2009 mit 12 Probanden statt. Hier waren die Ziele, die entwickelte Methodik und Modelle auf ihre Praxistauglichkeit hin zu untersuchen und ggf. Schwachstellen aufzudecken. Außerdem sollte durch den Vergleich mit einer Kontrollgruppe festgestellt werden, ob eine Verbesserung von Ergebnissen erreicht werden kann. Die Teilnehmer wurden daher in zwei Gruppen zu jeweils zwei unabhängigen Teams aufgeteilt. Zwei Teams erarbeiteten Anforderungslisten auf Basis der aus anderen Vorlesungen bekannten Methoden, z.B. Checklisten. Die anderen Teams wurden innerhalb eines Vormittags in das vorgestellte Modellierungskonzept (vgl. Ab-

schnitt 4) und die Anwendung der Software—soweit für das Projekt notwendig—eingewiesen.

5.1.4 Ergebnisse

Zur Beurteilung wurden zum einen die erzielten Entwicklungsergebnisse als objektive Bewertungsgrundlage herangezogen. Zum anderen wurden anonyme Befragungen mit Fragebögen durchgeführt, die das subjektive Empfinden der Teilnehmer erfassten. Der verwendete Fragebogen ist in Abb. C.1 (im Anhang) abgebildet. Es wurden 96% der Fragebögen ausgefüllt und zurückgeben (26 Bögen bei insgesamt 27 Probanden). Außerdem wurden die Eindrücke durch gezielte Gespräche mit den Teilnehmern hinterfragt und untermauert.

Stufe 1 Bei der Befragung der Teilnehmer wurde bereits in der ersten Stufe der Versuchsdurchführung eine gute Akzeptanz der Notation festgestellt. Die Grundzüge der Notation wurden innerhalb kurzer Zeit verstanden. Die Bedienung der Software wurde als in angemessener Zeit erlernbar bezeichnet. Lediglich einige Besonderheiten in der Benutzerführung sorgten anfänglich für Probleme. So war den Teilnehmern nicht intuitiv klar, dass durch die Löschung einer Visualisierung nicht auch das Objekt selbst im Modell gelöscht wird. Dieses Problem trat insbesondere bei Beziehung auf. Laut Selbsteinschätzung der Teilnehmer können Probleme dieser Art durch eine intensivere Einweisung ausgeräumt werden.

Die Anforderungsmodellierung erschien den Teilnehmern wichtig und sinnvoll. Insbesondere mussten sie sich systematisch mit ihren Zielen auseinandersetzen und wurden sich dabei über wichtige Zusammenhänge klar. Im Laufe des Projekts entschieden sich alle Teams von dem einstufigen Getriebe des Basismodells abzurücken und ein zweistufiges Getriebe zu entwickeln. Die erworbene Kenntnis der Zusammenhänge war insbesondere für die hierfür notwendigen Iterationen nützlich und kürzte den Entwicklungsprozess ab.

Die Hälfte der Teilnehmer war der Meinung, dass sie bereits durch ein intuitives Vorgehen ein sinnvolles Anforderungsmodell erstellen konnten. Ein Drittel meinte, dass ihnen durch die Anforderungsmodellierung wichtige Zusammenhänge klar wurden. Zwei Drittel äußerten, dass das Anforderungsmodell im vorliegenden Fall keine Zusatzinformation zur herkömmlichen Anforderungsliste bringen würde.

Als Grund für die letzte Aussage wird der begrenzte Rahmen des Projekts vermutet. Es handelt sich weder um eine sehr komplexe noch interdisziplinäre Problemstellung und bei einer kleinen Gruppengröße ist im Allgemeinen auch ohne Hilfsmittel eine ausreichende Kommunikation gewährleistet. Der Aufwand zum Erlernen der neuen Methode und der Anwendung des bisher unbekannten Rechnerhilfsmittels war in diesem Projekt höher als der erzielbare Nutzen. Daraus lässt sich schließen, dass ein gutes Nutzen zu Aufwand Verhältnis erst ab

einer gewissen Projektkomplexität der Tätigkeiten erreicht wird. Außerdem ist erst bei wiederholter Anwendung mit den positiven Effekten einer Lernkurve zu rechnen, wodurch das Nutzen zu Aufwand Verhältnis verbessert wird.

Stufe 2 In der zweiten Stufe konnten Unterschiede zwischen den Gruppen, die das entwickelte Konzept zur Anforderungsmodellierung nutzten, und den Kontrollgruppen festgestellt werden. Die Angaben der Fragebögen sind gemittelt in Abb. 5.2 graphisch dargestellt.

Nach ihrer eigenen Einschätzung waren die Teilnehmer der Kontrollgruppen deutlich weniger davon überzeugt, dass ihnen die Aufgabenklärung bei der Entwicklung geholfen hat (vgl. Abb. 5.2 a). Die Entwicklungsziele waren ihnen über große Strecken des Entwicklungsprozesses nicht vollständig klar. Die Teilnehmer der anderen Gruppen schätzten ihre Zielklärung und ihr Zielbewusstsein im Mittel etwa 16% besser ein. Offenbar erfolgte bei ihnen die Zielklärung zu Beginn der Entwicklung sogar so umfassend, dass spätere Ergänzungen des Anforderungsmodells nur selten vorgenommen wurden. In den Kontrollgruppen wurden hingegen vergleichsweise viele Änderungen und Eränzungen vorgenommen (nach eigenen Einschätzungen im Mittel etwa 18% mehr). Die Verteilung der Antworten war innerhalb der Kontrollgruppen allerdings wesentlich inhomogener. Die Anwendung der Anforderungsmodellierung lässt demnach einen homogeneren und damit planbareren Projektverlauf erwarten.

Untersucht man die in die verschiedenen Entwicklungsphasen investierte Zeit, so wurde der Großteil für die Auslegung verwendet (vgl. Abb. 5.2 b). Gut ein Viertel der jeweils investierten Gesamtzeit wurde für die Gestaltung und ein knappes Sechstel für die Aufgabenklärung verwendet. Insgesamt wendeten die Teilnehmer der Kontrollgruppe etwa ein Viertel weniger Arbeitszeit für die Bearbeitung auf.

Die Gruppen, die eine Anforderungsmodellierung durchführten, wurden weiterhin nach ihren Erfahrungen mit dem Modellierungskonzept und der Anwendung der Software befragt (vgl. Abb. 5.2 c). Sie gaben an, dass sie sich gut in die Anforderungsmodellierung haben einarbeiten können und dazu eine angemessene Zeit benötigten. Diese Aussagen decken sich mit denen des Vorjahres. Die Bedienbarkeit der verwendeten Software wurde—wie schon im Vorjahr—als verbesserungsbedürftig eingestuft. Die vorgestellte Methodik wurde trotz des vergleichsweise geringen Umfangs der Einweisung als hilfreich angesehen. Konsistent mit den vorherigen Aussagen wurde angegeben, dass die Modellierung dabei half wichtige Zusammenhänge zu erkennen. Die Teilnehmer waren diesmal durchweg der Meinung, dass das Anforderungsmodell gegenüber einer Anforderungsliste entscheidende Zusatzinformationen birgt.

Eine Repräsentation der Modellierungsergebnisse zeigt das Anforderungsdiagramm der Gruppe „*Green Speed Machine*“ in Abb. 5.3. Oben links zu erkennen ist das Zielsystem, aus dem sich, auf das Gesamtfahrzeug angewendet, z.B. Anforderungen wie *maximale Längsgeschwindigkeit* entwickeln. Mit dem Test-

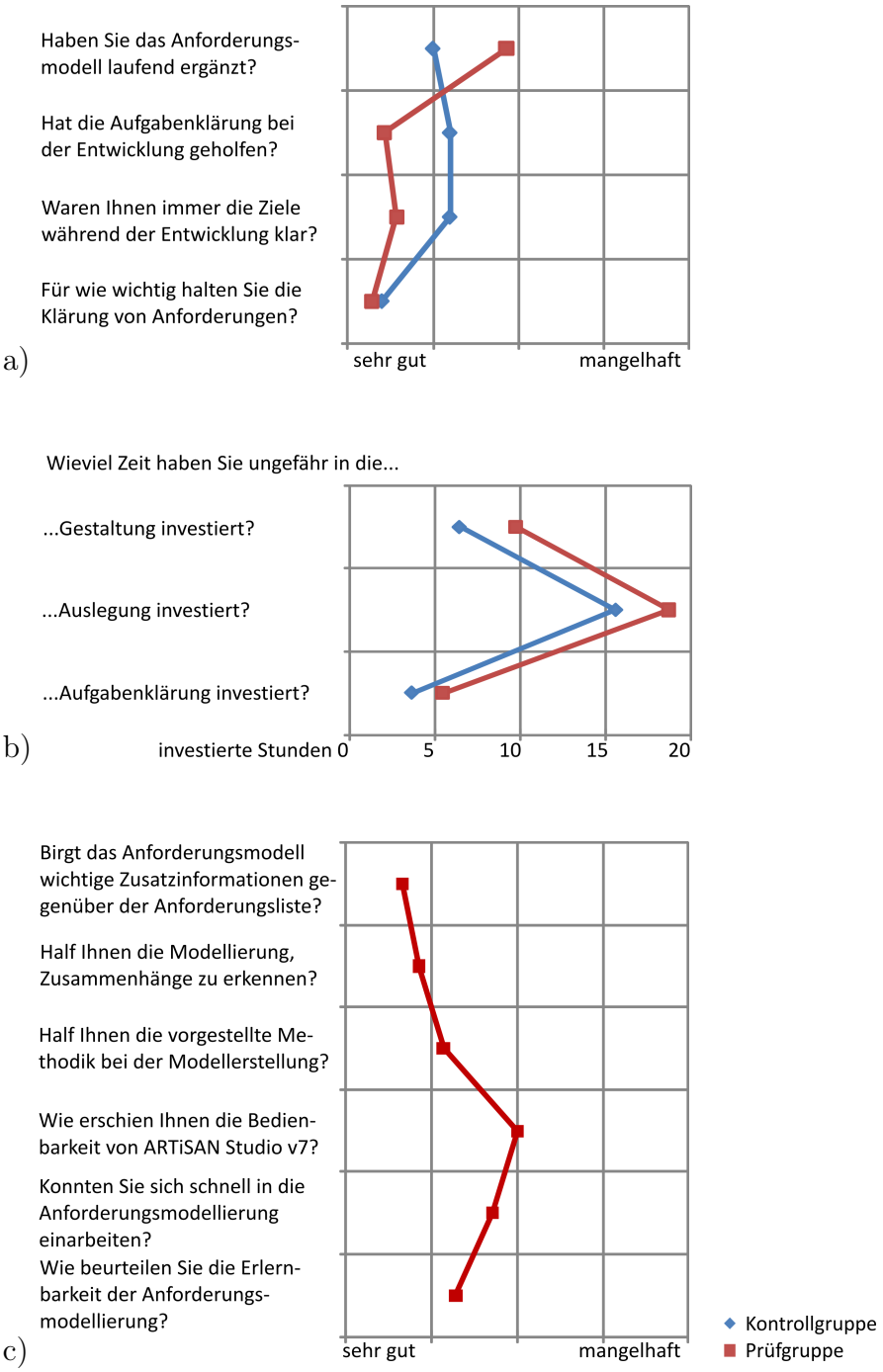


Abbildung 5.2: *Auswertung der Fragebögen im Wintersemester 2008/2009: a) Allgemeine Fragen zum Anforderungsmanagement in der Produktentwicklung, b) Zeitlicher Umfang bei der Produktentwicklung und c) Spezielle Fragen zur Anforderungsmodellierung mit erweiterter SysML-Notation.*

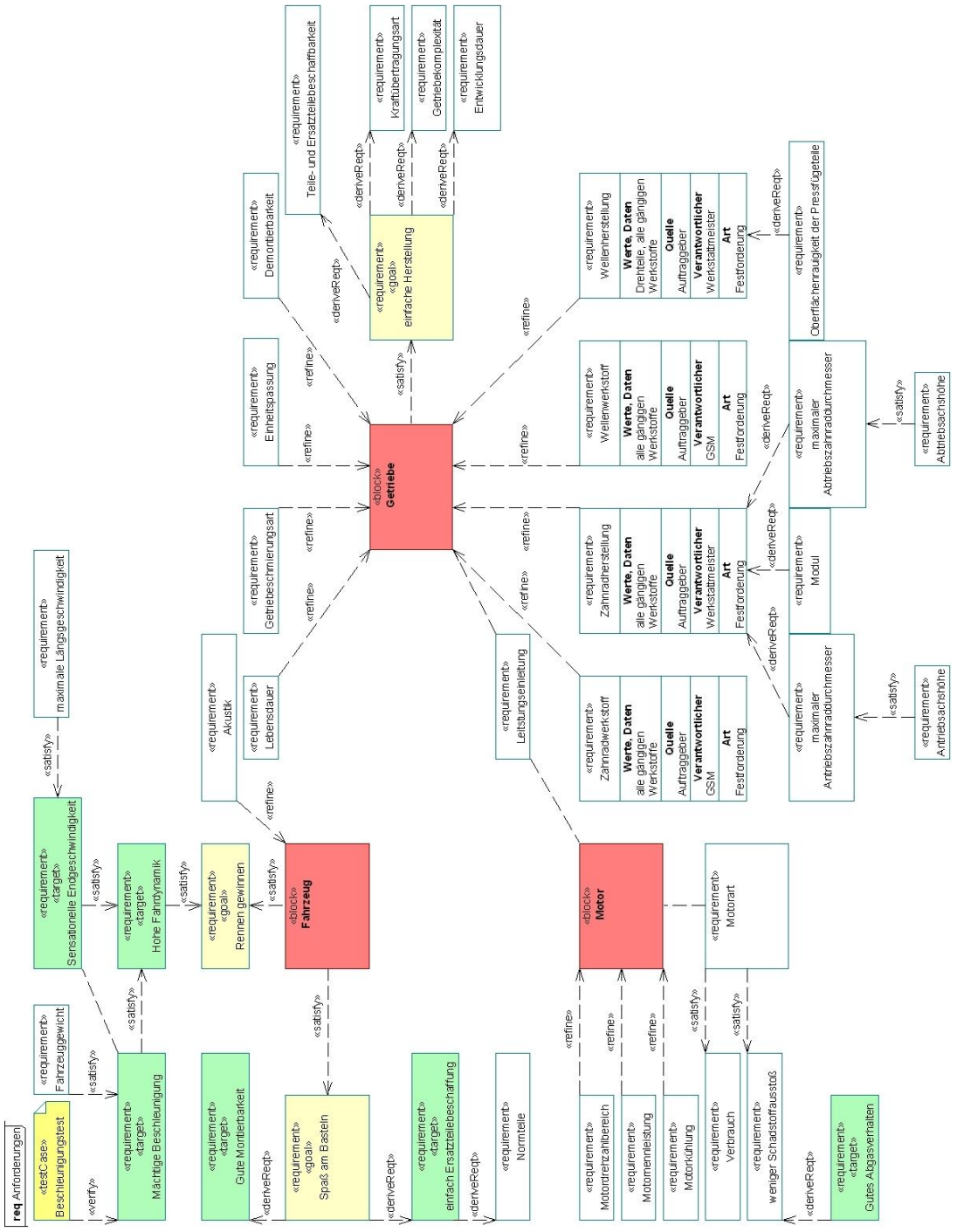


Abbildung 5.3: *Ein Anforderungsdiagramm der Arbeitsgruppe „Green Speed Machine“.*

fall *Beschleunigungstest* wurde berücksichtigt, wie die gestellten Anforderungen überprüft werden können. Am unteren Bildrand sind Restriktionen als Anforderungen dargestellt, die sich aus dem Fahrzeugpackage ergeben. Beispielsweise ist

die Achshöhe des Abtriebs nur durch sehr aufwändige Anpassungen am Getriebegehäuse und der Karosserie zu verändern. Daher wird sie als fest angenommen, um andere Anforderungen wie *kurze Entwicklungsdauer*—am rechten Bildrand dargestellt—nicht zu beeinträchtigen.

5.1.5 Fazit

Zwar sind die Modelle aus diesen ersten Anwendungstests noch verbesserungsbedürftig, insgesamt lässt sich aber feststellen, dass die Anforderungsmodellierung auf Basis der SysML von den Studierenden als hilfreich angenommen wurde. Auch der notwendige Zeitbedarf zur Erlernung der Modellierung ist, selbst bei wenig Vorkenntnissen, gering. Insgesamt war der in die Bearbeitung investierte Zeitanteil bei den Gruppen der Anforderungsmodellierung etwas höher, allerdings waren auch die erzielten Ergebnisse etwas besser. Beispielsweise wurden von den Kontrollgruppen im Wesentlichen Restriktionen aufgelistet, die für das konkrete Projekt jedoch häufig irrelevant waren (z.B. Innendurchmesser der Kupplungsglocke). Konzeptrelevante Anforderungen wurden—wenn überhaupt—nur so qualitativ beschrieben, dass sie nicht für die Entwicklung verwendet werden konnten (z.B. hohe Kurvengeschwindigkeit). In den anderen Gruppen wurden im Wesentlichen quantitative Anforderungen aufgestellt und es wurde ein wesentlich breiterer Bereich abgedeckt (z.B. gute Ersatzteilbeschaffbarkeit). Daraus resultierten konkrete Überlegungen z.B. für ein sinnvolles Fertigungsverfahren und schließlich die damit verbundenen Fertigungsrestriktionen und -möglichkeiten.

Es ist nicht mit Sicherheit zu sagen, ob das bessere Verständnis der Aufgabe zu einer höheren Motivation, einer höheren Arbeitsbereitschaft und somit schließlich zu besseren Arbeitsergebnissen geführt hat. Die Aspekte Mitarbeiter (Motivation, Fachwissen), Arbeitstechnik/Hilfsmittel und Organisation bedingen sich gegenseitig [Bir91]. Die Modellierung hat jedoch mit Sicherheit subjektiv ein besseres Verständnis der Aufgabenstellung gefördert und dabei nur einen vertretbaren Mehraufwand gefordert.

Eine allgemeingültige Aussage über den Nutzen kann nur durch weitere Untersuchungen—insbesondere unter Einbeziehung von erfahrenen Entwicklern—erreicht werden. Wegen der geringen Anzahl von Probanden kann nicht ausgeschlossen werden, dass zufällig besonders gute Studenten in der Prüfgruppe waren. Außerdem lassen sich Untersuchungen mit Studenten nicht ohne weiteres auf erfahrene Entwickler übertragen. Diese ersten Stichproben zeigen aber deutlich, dass das Vorgehen zu einer Verbesserung der Produktentwicklung beitragen und dass eine hohe Akzeptanz erwartet werden kann. Die Einführung in das Anforderungsmanagement mit SysML ist daher inzwischen fester Bestandteil der Vorlesung.

5.2 Modellierung von Parallelrobotern

Der im Jahre 2000 gestartete Sonderforschungsbereich (SFB) 562 „Robotersysteme für Handhabung und Montage—Hochdynamische Parallelstrukturen mit adaptronischen Komponenten“ befasst sich unter anderem mit Entwicklungsmethoden für Parallelroboter. In Abb. 5.4 ist die Struktur des SFB abgebildet, d.h. welche Teilprojekte bearbeitet werden und wie diese zusammenwirken. Die verschiedenen Teilprojekte sind in die Themenbereiche *A–Entwurf und Modellierung*, *B–Steuerung und Informationsverarbeitung* und *C–Neue Komponenten* eingeteilt. Die beschriebene Modellierung wurde im Rahmen der Arbeiten des Teilprojekts A1 „*Wissensunterstützte Entwicklungsmethoden und modulares Baukastenkonzept*“ entwickelt.

In Abb. 5.5 ist ein allgemeiner Funktionsplan für Industrieroboter gezeigt. Über das Bediensystem und die Programmierumgebung werden die Bahnplanung und Robotersteuerung bedient. Aus der Robotersteuerung folgen die Sollwerte für die Aktorstellglieder, die auf die Antriebe bzw. den Effektor wirken. Die Antriebe leiten eine Bewegung des Bewegungswandlers ein. Durch Bewegungswandler und Effektor werden Objekte manipuliert. Durch die Sensorik werden externe (z.B. Positionierung) und interne (z.B. Gelenkwinkel) Messwerte an die Robotersteuerung zurückgegeben.

Der Bewegungswandler kann als serielle, parallele oder hybride kinematische Struktur ausgeführt werden. Abb. 5.6 a) zeigt eine offene kinematische Kette zum Aufbau eines seriellen Roboters. Hier sind sämtliche Glieder durch aktive Gelenke (d.h. Antriebe) miteinander verbunden. Das erste Gelenk verbindet die kinematische Kette mit dem Gestell, das letzte Gelenk mit dem Endeffektor.

Bei der geschlossenen kinematische Kette in Abb. 5.6 b) verbindet das erste Gelenk ebenfalls die Struktur mit dem Gestell. Das letzte Gelenk verbindet die Kette mit der Arbeitsplattform auf der sich der Effektor befindet. Da an der Arbeitsplattform mindestens zwei kinematische Ketten angebunden sind, entsteht eine geschlossene Kette. Innerhalb der Struktur werden aktive Gelenke gestellnah angeordnet. Die übrigen Gelenke sind passiv, d.h. nicht angetrieben.

Wird—wie in Abb. 5.6 c) gezeigt—innerhalb der parallelen Grundstruktur ein zusätzliches aktives Gelenk eingebracht, so spricht man von einer hybrid-parallel-seriellen Struktur. Wird in eine serielle Grundstruktur ein paralleler Zweig eingebracht, so spricht man von hybrid-seriell-parallelen Strukturen. Hybridstrukturen werden genutzt, um die Vorteile von parallelen und seriellen Strukturen zu kombinieren bzw. die Komplexität zu reduzieren.

Abb. 5.6 d) zeigt eine redundante Parallelstruktur. Hier wurde eine kinematische Kette verdoppelt, um z.B. Singularitäten zu durchfahren oder höhere Steifigkeiten durch Verspannung zu erreichen.

Die systematische Beschreibung von parallelkinematischen Strukturen basiert auf der Hintereinanderschaltung von Gelenken verschiedener Freiheitsgrade und Bewegungsrichtungen. Dabei hat sich die englischsprachige Notation gegenüber

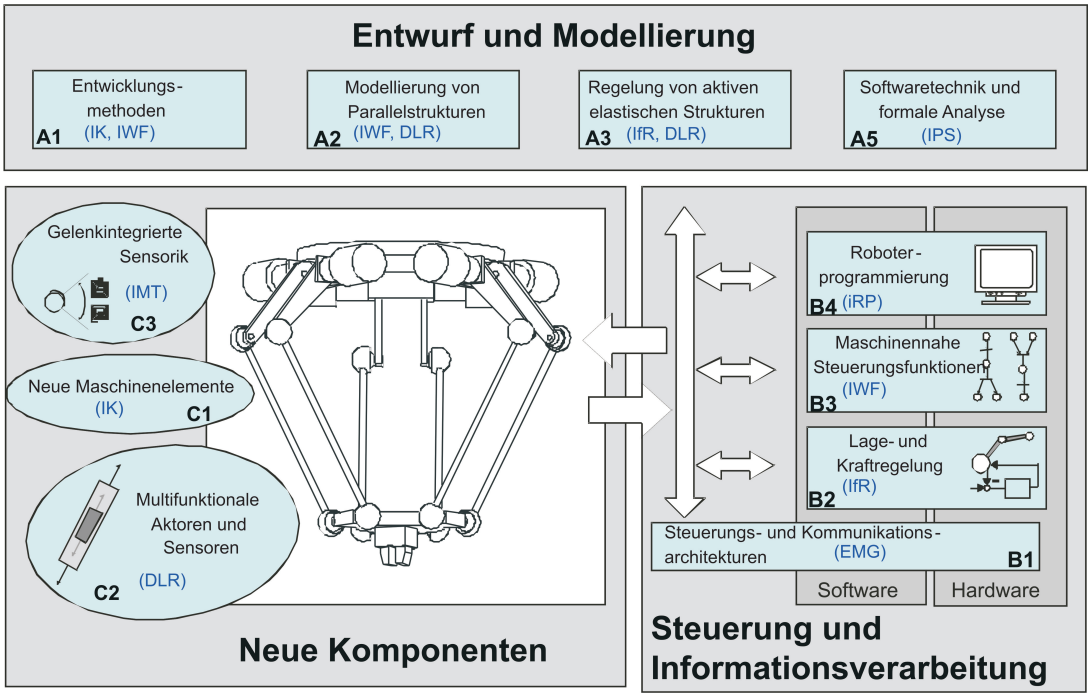


Abbildung 5.4: *Aufbau und Zusammenwirken der verschiedenen Teilprojekte im SFB 562 „Robotersysteme für Handhabung und Montage“, nach [Wah06, S. 25].*

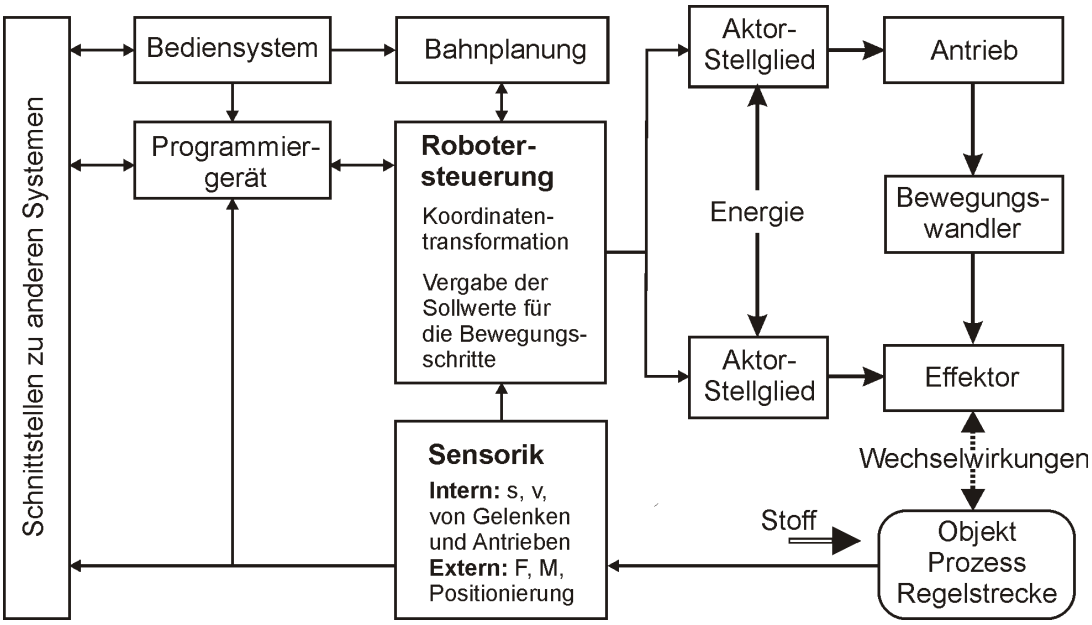


Abbildung 5.5: *Allgemeiner Funktionsplan für Industrieroboter, nach [Czi06, S. 140f].*

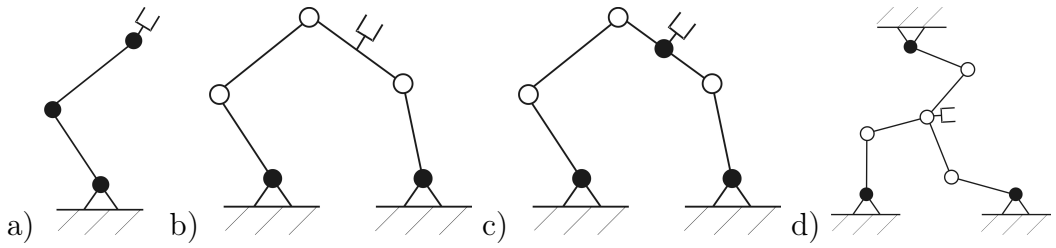


Abbildung 5.6: *Unterschiedlicher Aufbau eines Bewegungswandlers: a) seriell, b) parallel, c) hybrid und d) redundant (○ passives Gelenk, ● aktives Gelenk (Antrieb)).*

der in VDI-RL-2156 [VDI07] oder bei *Otremba* [Otr05] beschriebenen international durchgesetzt, z.B.

- **R** - rotational (D - Drehgelenk mit einem rotatorischen Freiheitsgrad)
- **P** - prismatic (S - Schubgelenk mit einem translatorischen Freiheitsgrad)
- **U** - universal (D_2 - Kreuzgelenk mit zwei rotatorischen Freiheitsgraden)
- **S** - spherical (D_3 - Kugelgelenk mit drei rotatorischen Freiheitsgraden)

Die Beschreibung erfolgt ausgehend vom Gestell zur Arbeitsplattform. Bei identischem Aufbau der Ketten wird die Anzahl der Ketten der Kettenbeschreibung vorangestellt. Das aktive Gelenk wird durch Unterstreichung gekennzeichnet. **6-RUS** bezeichnet demnach eine Struktur vom Typ Hexa (vgl. Abb. 5.4), d.h. sechs identische kinematische Ketten mit jeweils einem Drehantrieb, einem Kardangelenk und einem Kugelgelenk. **RRRRR** bezeichnet ein Fünfgelenk mit zwei im Gestell angeordneten Drehantrieben und drei passiven rotatorischen Gelenken, wie es in Abb. 5.6 b) dargestellt ist.

Parallelkinematische Roboter verfügen gegenüber seriellkinematischen Robotern über einige Vorteile. Sie besitzen bei gleichem Materialaufwand eine höhere Steifigkeit, da durch die geschlossenen kinematischen Ketten die Arbeitsplattform mehrfach abgestützt wird [Bud09, S. 3]. Außerdem werden die vergleichsweise schweren Antriebe im Gestell oder zumindest gestellnah platziert. Dadurch verringern sich die bewegten Massen bzw. die Trägheitsmomente [Ker98]. Des weiteren können Strukturen derart aufgebaut werden, dass die Glieder überwiegend auf Zug und Druck belastet werden. Dadurch können die bewegten Massen weiter verringert werden [Tre98]. Die geringeren bewegten Massen führen zu einer höheren realisierbaren Dynamik, d.h. es sind höhere Beschleunigungen des TCP möglich [Bud09, S. 3]. Nach *Frindt* [Fri01] lassen sich unterschiedliche parallelkinematische Strukturen durch wenige unterschiedliche Komponenten aufbauen. Daher kann hier bei einem relativ geringen Umfang an internen Varianten eine große externe Varianz erzeugt werden [Ste06b]. Durch eine konsequente Verwendung von Gleichteilen können zudem Kosten reduziert werden [Fra02c].

Nachteile eines Parallelroboters sind das relativ kleine Verhältnis von Arbeitsraum zu Bauraum. Der Arbeitsraum ist maximal so groß, wie die Schnittmenge der Einzelarbeitsräume jeder kinematischen Kette [Fri01]. Zudem liegen innerhalb des Arbeitsraums häufig Singularitäten des Typs 2 vor. In singulären Posen ist die Struktur durch die Antriebe nicht kontrollierbar. Es müssen Sicherheitsabstände zu Singularitäten eingehalten werden, wodurch der nutzbare Arbeitsraum noch kleiner wird [Bie06]. Darüberhinaus ist bei Strukturen mit Rotationsfreiheitsgraden am TCP die Orientierbarkeit häufig stark eingeschränkt [Kre06a]. Weitere Schwierigkeiten ergeben sich durch die stark posenabhängigen kinematischen Eigenschaften [Tre98], den erhöhten Aufwand bei der kinematischen und dynamischen Modellierung [Pis00] und bei der Steuerungsauslegung [Hes00].

Als ein komplexes mechatronisches Produkt sind an der Entwicklung von Parallelrobotern verschiedene technische Disziplinen beteiligt und eine Reihe unterschiedlicher Partialmodelle notwendig [Ste07a]. Hierbei sind die komplexen Beziehungen zwischen den verschiedenen Disziplinen zu berücksichtigen. Bisher werden Parallelroboter nicht als „Stangenware“ verkauft, sondern stets an die Randbedingungen eines speziellen Kunden angepasst. Die Wiederverwendung von Wissen, die Möglichkeit der Konfiguration und Rekonfiguration aus einem Baukasten und ein effektives Änderungsmanagement durch eine ganzheitliche Sichtweise auf den Roboterlebenslauf würden es ermöglichen, schnelle Markteinführungszeiten bei hoher Qualität und sicherer Erfüllung der Kundenwünsche zu gewährleisten.

Im Bereich Parallelroboter sind nur in geringem Umfang Ansätze für Baukastensysteme zu finden. Vorhandene Ansätze beschränken sich auf bestimmte Strukturen oder Teile des Entwicklungsprozesses (z.B. Stabkinematiken in [Pri97b] [Pri97a]). Portalroboter wurden in [Gör01] und serielle Strukturen in [Jun88] [Tsc00] [Yan00] betrachtet. Allgemeine Vorgehensweisen, um Parallelroboter anforderungsgerecht abzubilden und dabei eine Vielzahl von unterschiedlichen parallelen Strukturen zu ermöglichen, fehlen. Ein solcher Ansatz muss die Anforderungen als Startpunkt ansehen und den gesamten Entwicklungsprozess bis hin zur Definition von Modulen und Schnittstellen begleiten. Dabei muss das Vorgehen für verschiedene Strukturen, Anwendungsfälle und Umgebungen flexibel einsetzbar sein.

Das in Abschnitt 4 entwickelte Konzept hilft durch eine ausführliche Modellierung des Zielsystems und der Anforderungen die Kundenwünsche und gleichzeitig die unterschiedlichen Unternehmensziele auch bei kooperativer Produktentwicklung zu treffen. Im Folgenden wird am Beispiel gezeigt, wie mit diesem Ansatz die Entwicklung von Baukastensystemen unterstützt, Parameter zur Rekonfiguration ausgemacht und geeignete Testfälle identifiziert werden können. Dabei folgt die Beschreibung im Wesentlichen der in Abb. 4.1 dargelegten Abfolge. Da die Modellierung, wie schon beschrieben, nicht streng sequenziell erfolgen kann (vgl. Abschnitt 2.1), werden in den folgenden Abschnitten z.T. Vorgriffe auf im Ablauf erst später angegebene Modelle gemacht. Dieses entspricht etwa der realen Arbeitsweise. Ein Partialmodell wird immer dann weiter detailliert, wenn die De-

taillierung an einer anderen Stelle es nötig macht. Beispielsweise werden bisher nicht berücksichtigte Stakeholder dann eingefügt, wenn sie als Akteure für neue Anwendungsfälle gebraucht werden.

5.2.1 Idee für ein Handhabungs- und Montagesystem

Zu Beginn der Entwicklung steht die Systemidee, die den Wesenskern des Systems beschreibt (vgl. Abschnitt 2.1.1 und Abschnitt 4.1.1). Sie wird möglichst allgemein formuliert und dient zur Strategiebildung und Abgrenzung von anderen alternativen Produktarten. Im vorliegenden Fall wird die Systemidee durch sechs Objekte dargestellt:

- breiter Anwendungsbereich
  Das System soll sich für ein breites Spektrum von Anwendungen eignen. Es sollen also verschiedenartige Handhabungs- und Montageaufgaben unterschiedlicher Branchen mit dem gleichen System ausgeführt werden können.

- hohe Flexibilität
  Das System soll während des Betriebes flexibel an andere Aufgaben anpassbar sein.

- hohe Qualität
  Das System soll eine hohe Qualität bei der Handhabung und Montage des produzierten Gutes sicherstellen.

- hoher Ausstoß
  Das System soll in möglichst wenig Zeit einen möglichst hohen Ausstoß liefern, d.h. es soll eine möglichst große Anzahl an Wiederholungen der Handhabungs- oder Montageaufgabe in einem möglichst kurzen Zeitraum durchgeführt werden.

- hoher Gewinn
  Das System soll dazu beitragen, dass ein hoher Unternehmensgewinn erzielt wird.

- hohes Ansehen
  Das System soll dazu beitragen ein hohes Ansehen des Unternehmens in der Gesellschaft (insb. beim Kunden) zu installieren bzw. zu halten.

Die Objekte werden wie in Abb. 5.7 a) im Packagebrowser dargestellt im Modell dokumentiert. Somit sind sie für alle Projektbeteiligten sichtbar und können im weiteren Verlauf zur Ausarbeitung der Ziele und zur Modellauswertung verwendet werden. In Abb. 5.7 b) sind die Merkmale im Stereotyp `<<goal>>` für das Objekt „*breiter Anwendungsbereich*“ gezeigt.

Wird an dieser Stelle nur die Systemidee betrachtet, so sind eine Reihe sehr

a) 

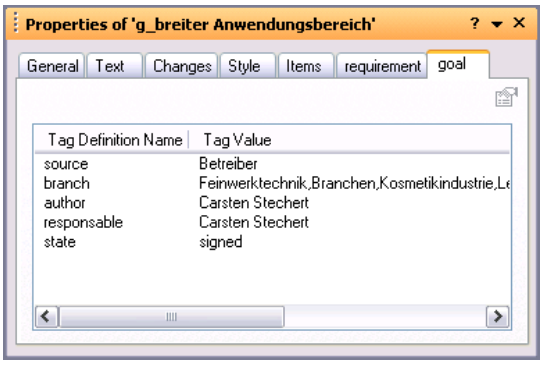
b) 

Abbildung 5.7: *Dokumentation der Idee für ein Handhabungs- und Montagesystem im Modell: a) Objekte im Packagebrowser und b) Merkmale eines Objekts im Propertiesbrowser.*

unterschiedlicher Lösungen denkbar. Das fertig entwickelte System muss sich auch an diesen Alternativen messen können. Eine mögliche Alternative ist die Handhabung und Montage durch Menschen. Diese Lösung kann ein hohes Ansehen bewirken, z.B. falls im eigenen Land produziert wird. Zwar können sich Menschen im Prinzip schneller auf vollständig neue Aufgaben einstellen als Robotersysteme, bei häufig wechselnden oder einer hohen Vielfalt der Aufgaben sind Robotersysteme hingegen flexibler einsetzbar (schnelleres „Umschalten“, keine Lernkurve). Einem hohen Ausstoß (durch viele Arbeiter) stehen hohe Personalkosten gegenüber, die den Gewinn schmälern.

Automaten, als weitere Alternative, können die Flexibilität nicht erfüllen. Serielle Roboter erreichen, verglichen mit Parallelrobotern, aufgrund geringerer Dynamik nicht den gewünschten hohen Ausstoß.

5.2.2 Ziele für die Entwicklung eines Parallelroboters

Die Überlegungen zur Systemidee lassen folgern, dass Parallelroboter bei den betrachteten Randbedingungen am ehesten geeignet sind. Das Zielobjekt ist demnach „Parallelroboter“ und wird als `<<block>>` im Package „Systemkontext“ erstellt (vgl. Abb. 5.8 a) unten).

Die Zielgruppe wird im Stakeholdermodell durch Rollen unterhalb des Kunden modelliert (s. Abschnitt 5.2.5). Beispielsweise sind Produktentwickler auf Kundenseite Lebensmitteltechnologen oder Chemiker. Zusammen mit den Anwendungsfällen im Package Verwendung des Produktlebenslaufs (z.B. Verpacken von Backzusatzstoffen, flexibles Befüllen von Gel-Permeations-Chromatographie (GPC)-Säulen, s. Abschnitt 5.2.3) ergibt sich ein Überblick über das abzudeckende Marktsegment.

Der Zweck des Parallelroboters wird durch Ziele genauer beschrieben. Diese werden im Modell dokumentiert. In Abb. 5.8 b) ist im Propertiesbrowser die Be-

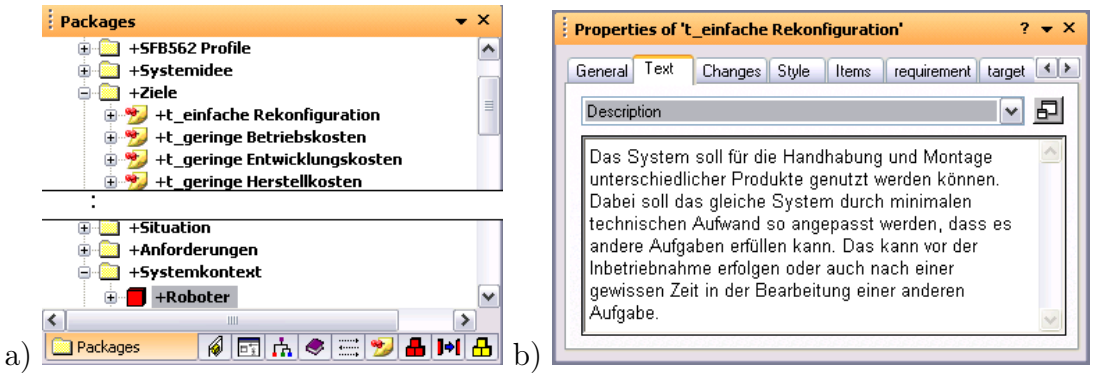


Abbildung 5.8: *Dokumentation der Ziele für einen Parallelroboter im Modell: a) Objekte im Packagebrowser und b) Beschreibung eines Ziels im Propertiesbrowser.*

schreibung für das Ziel „*einfache Rekonfiguration*“ abgebildet.

Die Ziele erfüllen die Objekte der Systemidee, wobei es sich nicht um 1:1-Zuordnungen handelt. Das Ziel „*kurze Einarbeitungszeit*“ trägt beispielsweise zur Erfüllung aller Objekte der Systemidee bei. Andersherum trägt etwa die Hälfte aller aufgestellten Ziele zur Erfüllung der hohen Flexibilität bei. Diese Zusammenhänge lassen sich in einem Anforderungsdiagramm und in einer *Satisfy*-Matrix (vgl. Ausschnitt der Matrix in Abb. 5.9) übersichtlich darstellen. Beide Darstellungen werden im Modell dokumentiert (vgl. Abb. 5.7 a).

An dieser Stelle werden geeignete Strategien zur Zielerfüllung festgelegt. Wie schon in Abb. 4.4 gezeigt wurde, ist eine Baukastenstrategie geeignet die Zielerfüllung der dort dargestellten Ziele zu unterstützen. Beispielsweise wer-

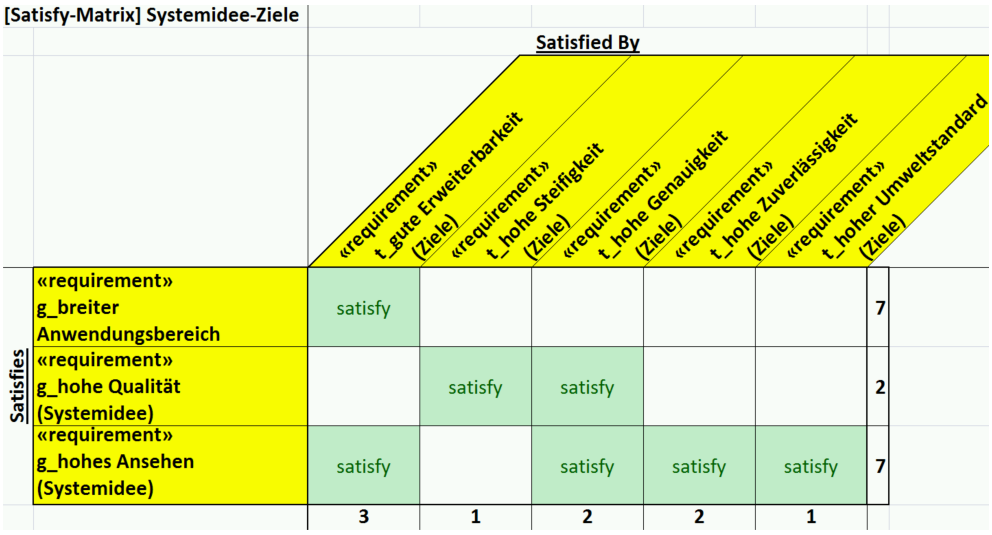


Abbildung 5.9: *Die Satisfy-Matrix gibt die Erfüllungsbeziehungen zwischen den Zielen und den Objekten der Systemidee übersichtlich in einem Excel-Arbeitsblatt an.*

den die Herstell- und Entwicklungskosten der Varianten durch die Verwendung von Gleichteilen reduziert. Gleichzeitig unterstützt die Baukastenstrategie weitere Ziele, die bisher (*top-down* Sichtweise) nicht betrachtet wurden. Mit Hilfe des Baukastensystems werden z.B. Varianten schneller entwickelt, d.h. die Markteinführungszeit wird verkürzt. Dieses Ziel wird zusätzlich in das Modell aufgenommen, da es geeignet ist das „*hohe Ansehen*“ der Systemidee zu erfüllen (*bottom-up* Sichtweise).

Neben der Sicherstellung der Durchführbarkeit des speziellen Anwendungsfalls müssen also auch die Objekte der Systemidee berücksichtigt werden. Beispielsweise soll ein hoher Ausstoß erreicht werden. Es gibt einige Ziele der darunterliegenden Ebene durch die dieses gestützt wird. Allerdings befinden sich nicht immer alle innerhalb der betrachteten Systemgrenzen. Beispielsweise soll eine optimierte Zuführung zum Roboter hier nicht betrachtet werden. Das Ziel, eine hohe Dynamik zu erzielen, ist hingegen ein direkter Unterstützer und kann z.B. durch die technischen Anforderungen *hohe Beschleunigung am TCP erzielen* und *hohe Geschwindigkeit am TCP erzielen* ausgedrückt werden.

5.2.3 Produktlebenslauf eines Parallelroboters

Für die Modellierung des Produktlebenslaufes des Parallelroboters werden alle relevanten Anwendungsfälle vor, während und nach der Nutzung erfasst und dokumentiert (vgl. Abschnitt 4.1.3). Dabei geht es zunächst nicht darum die Anwendungsfälle möglichst detailliert zu beschreiben, sondern den Produktlebenslauf möglichst vollständig zu erfassen. Da es sich bei Parallelrobotern um Investitionsgüter hoher Lebensdauer handelt sind die Anwendungsfälle „nach der Nutzung“ vergleichsweise unsicher. Es ist kaum vorherzusagen, welche gesetzlichen Vorschriften gelten oder welche Verwertungstechnologien verfügbar sind, wenn nach 10 oder 20 Jahren die maximale Lebensdauer des Roboters erreicht ist.

Für die weitere Entwicklung und insbesondere für die Entwicklung des Baukastensystems sollen die Anwendungsfälle der Verwendung genauer betrachtet werden. Dazu werden die Anwendungsfälle, die zum Teil schon bei der Festlegung der Zielgruppe und des Marktsegments modelliert wurden (s. Abschnitt 5.2.2), strukturiert (vgl Abb. 5.10). Der im SFB562 üblichen Unterscheidung folgend wird sukzessive in die Aufgabenklassen Handhabung und Montage und weiter in Pick-and-Place, Verpackung, Beschickung, sowie flexible Kleinteilmontage, flexible Montage und Montage mit hoher Präzision unterschieden. Als weitere Spezialisierungen der Aufgabenklassen stehen vergleichsweise konkrete Anwendungsfälle, wie z.B. „*Muffins sortieren*“ oder „*GPC-Säulen befüllen*“.

Die Anwendungsfälle werden mit den jeweils relevanten Stakeholdern verbunden. Es ergibt sich eine breite Aufstellung von Anwendungen und Bereichen, in denen Varianten des aus dem Baukasten entwickelten Roboters eingesetzt werden sollen.

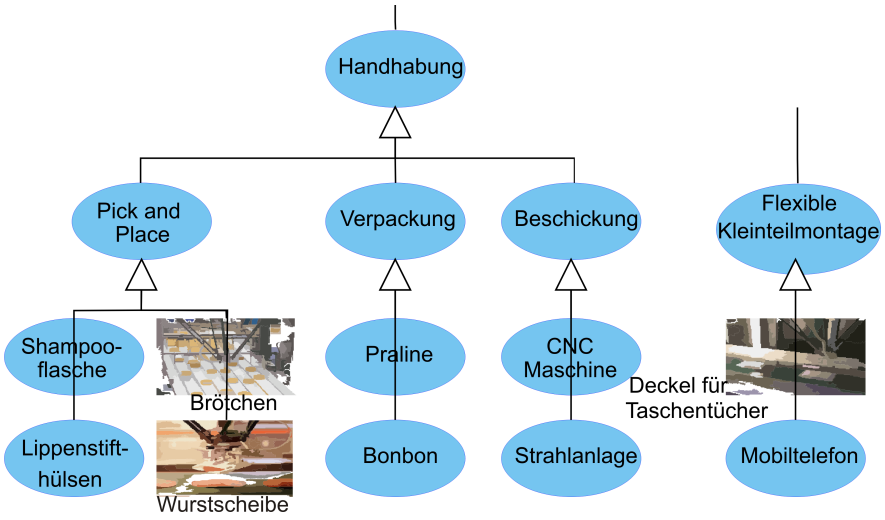


Abbildung 5.10: *Ausschnitt aus dem Anwendungsfalldiagramm zur Strukturierung der möglichen Aufgaben des Robotersystems, teilweise mit hinterlegtem Foto des Anwendungsfalls.*

5.2.4 Produktumgebung eines Parallelroboters

Die Produktumgebung eines Parallelroboters wurde bereits in Abb. 4.7 gezeigt. Dort sind auf Anlagenebene die bei der Produktentwicklung zu berücksichtigenden Objekte und ihre (potenziellen) Schnittstellen zum Roboter und untereinander dargestellt. Beispielsweise handhabt der Roboter ein oder mehrere (`1..*`) Bearbeitungsobjekte. Diese können unterschiedliche Geometrien, stoffliche Eigenschaften und Massen aufweisen. Insbesondere für den Greifer macht es einen Unterschied, ob eine vergleichsweise leichte Handyschale mit bestimmter, bekannter Geometrie gegriffen wird oder ein vergleichsweise schwerer Kuchen, der zudem nachgiebig ist. Je nachdem, welche Anwendungsfälle gleichzeitig zu betrachten sind (vgl. Abschnitt 5.2.3) muss hier z.B. ein flexibles Greifersystem oder ein Greifermodul mit Wechseleinrichtung realisiert werden.

Die Bearbeitungsobjekte werden entweder über Behälter (Magazin) oder Fördermittel dem Roboter zugeführt. In Abb. 5.11 ist eine Einteilung der zur Verfügung stehenden Fördermittel gezeigt. Durch die Entscheidung für eine bestimmte Zuführung ergeben sich zum Teil unterschiedliche Randbedingungen für die Realisierung. Beispielsweise bauen Gurtbahnen relativ flach und können fast in beliebiger Höhe an das Robotergestell herangeführt werden. Zudem versorgen sie den Roboter stetig mit Bearbeitungsobjekten, d.h. Ausfallzeiten des Robotersystems (z.B. auf Grund von Kalibrierungsmaßnahmen) führen sofort zu einem Produktionsrückgang. Bei fahrerlosen Transportsystemen (als Unstetigförderer) hingegen können systembedingte „Pausenzeiten“ in der Zuführung zur Kalibrierung genutzt werden. Im ersten Fall ist demnach ein „online“-Kalibrierungsverfahren (z.B. durch Nutzung strukturintegrierter Sensoren [Büt05] oder durch gezielte

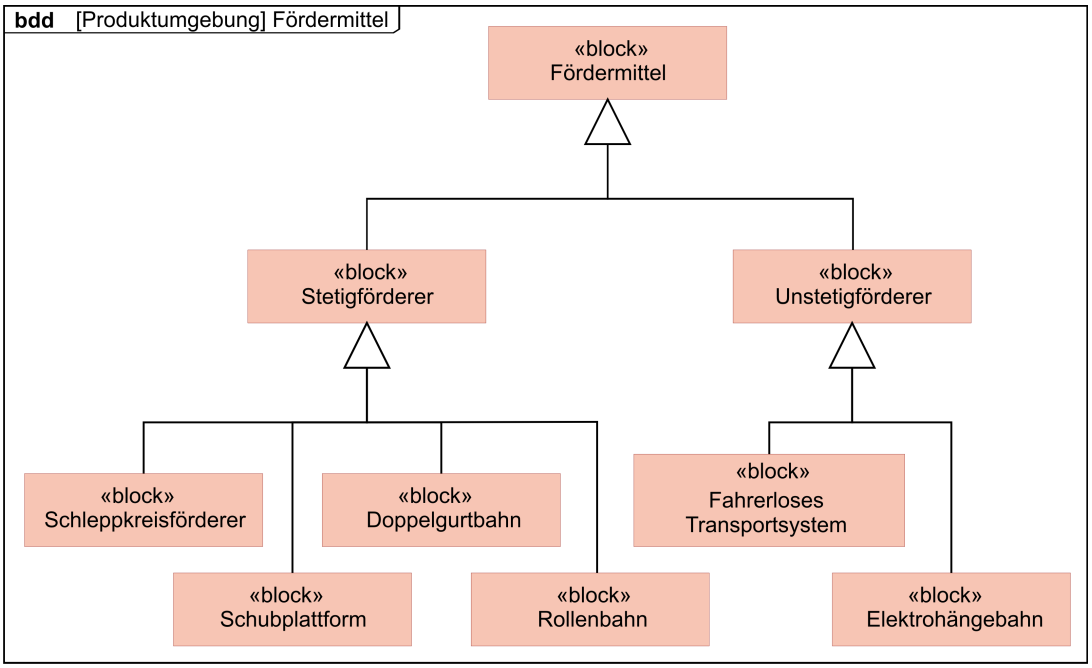


Abbildung 5.11: *Darstellung der zu betrachtenden alternativen Fördermittel für eine Zuführung der Handhabungsobjekte in den Arbeitsraum des Parallelroboters.*

Einschränkung von Bewegungsfreiheiten [Las08]) anzustreben. Im zweiten Fall sind Verfahren wie z.B. Konturverfolgung möglich. Hier steht der Roboter jedoch während der Kalibrierung nicht für seine eigentliche Aufgabe zur Verfügung.

Die eben genannten Restriktionen bzw. auf das Robotersystem zurückwirkenden Anforderungen werden im Modell dokumentiert und den entsprechenden Objekten der Produktumgebung zugeordnet (`allocate`). Falls eine Entscheidung getroffen wird, die z.B. zum Ausschluss von fahrerlosen Transportsystemen als Fördermittel führt, werden sämtliche daraus resultierenden Restriktionen und Anforderungen aus dem Modell entfernt und nicht mehr für die weitere Entwicklung berücksichtigt.

5.2.5 Stakeholder der Parallelroboterentwicklung

In Abb. 4.8 wurde bereits ein Ausschnitt aus der Stakeholderstruktur für Parallelroboter vorgestellt. In dem keineswegs vollständigen Modell für die Parallelroboterentwicklung innerhalb des SFB 562 sind derzeit über einhundert Stakeholder abgelegt. Davon sind 35 natürliche Personen, d.h. Projektleiter, wissenschaftliche Mitarbeiter und Studenten. Eine Auswertung der Stakeholder-Anwendungsfälle-Matrix (vgl. Abschnitt 4.2.2) ergibt, dass die Entwickler und dabei insbesondere die Konstrukteure an einem Großteil der Anwendungsfälle beteiligt sind. Außerdem zeigt sich, dass Anwender, Fertiger und Kostenanalytiker bereits früh in die

	A	B	C	D	E	G	H	I	J
1	Anwendungsfälle-Stakeholder-Matrix					Vor der Nutzung			
2							Entwicklung		Verkauf
3							Aufgabenklärung		
4									
5					Lebenslaufphasen				
6					Anwendungsfälle	Anforderungen aufbereiten	Anforderungen bereitstellen	Anforderungen erfassen	Anfrage
7	Ebene 1	Ebene 2	Ebene 3	Ebene 4	Stakeholder der Ebene 5				
98	Hersteller	Verwaltung	Manager	Projektleiter	Friedrich Wahl				
99					Hans-Joachim Franke				
100					Harald Michalik				
101					Jürgen Hesselbach				
102					Michael Sinapius				
103					Stephanus Büttgenbach				
104					Thomas Vietor				
114					Summe beteiligter Stakeholder	1	39	51	7

Abbildung 5.12: *Ausschnitt aus einer Anwendungsfälle-Stakeholder-Matrix für Parallelroboter.*

Entwicklung eingebunden werden müssen.

Abb. 5.12 zeigt einen Ausschnitt der durch die entwickelten Makros automatisch erzeugten Anwendungsfälle-Stakeholder-Matrix. In der letzten Zeile sind für jeden Anwendungsfall die Anzahl beteiligter Stakeholder angezeigt. Etwa die Hälfte aller beteiligten Stakeholder (51 von 106) werden für das Erfassen von Anforderungen eingebunden. An der Bereitstellung der Anforderungen sind weiterhin alle Entwickler und deren übergeordnete Rollen beteiligt.

Durch die gezielte Analyse einzelner Anwendungsfälle im erarbeiteten Modell zeigt sich, dass z.B. für eine Wartungsplanung nicht nur Wartungstechniker als Serviceanbieter befragt werden sollten, sondern ebenfalls Maschinenzustandsprüfer des Kunden und Kundenberater des Herstellers. Nur durch die Berücksichtigung aller Beteiligten wird eine vollständige Erfassung der Anforderungen möglich.

5.2.6 Anforderungen an Parallelroboter

Das erarbeitete Anforderungsmodell für Parallelroboter enthält über vierhundert Anforderungen, die sich auf die Systemebene und einzelne Komponenten beziehen. Dabei wurden Schwerpunkte auf einige maschinenbauliche Komponenten gesetzt (z.B. Gestell, Gelenke). Eine Auswertung der Anforderungen-Anwendungsfälle-Matrix (vgl. Abb. 4.18 in Abschnitt 4.2.3) führt zu der Erkenntnis, dass neben den speziellen Anforderungen, die aus den verschiedenen Anwendungen resultieren, ebenfalls ein großer Teil der Anforderungen durch Restriktionen der Fertigung und insbesondere der Montage auftreten. Weitere wichtige Anforderungen hängen mit der Möglichkeit zur Rekonfiguration des Robotersystems zusammen (vgl. Abschnitt 5.3).

Die Anforderungen, die von den meisten Anwendungsfällen adressiert werden, berücksichtigen das Handhabungsobjekt, Umgebungsbedingungen (z.B. Tempe-

ratur, Luftfeuchtigkeit) sowie Zugänglichkeit und Reinigungsmöglichkeiten. Wichtige funktionale Anforderungen betreffen die erreichbare Beschleunigung, Geschwindigkeit und Nennlast.

Unter Zuhilfenahme der Suchfunktion von Artisan Studio erhält man eine Liste sämtlicher *rationale*, mit denen Anforderungsspreizungen im Modell kenntlich gemacht wurden (s. Abb. 5.13). Untersucht man sämtliche Anforderungsspreizungen (vgl. Abschnitt 2.3.1 und Abschnitt 4.3), so erhält man zunächst eine Anzahl an Anforderungen, die durch Harmonisierung auf einen Wert festgelegt werden können (z.B. zulässige Lärmbelästigung, Wartungshäufigkeit, Garantie). Des Weiteren erhält man Anforderungen, die auf Grund spezieller Anwendungsbereiche so unterschiedlich sind, dass die Realisierung mehrerer Produktlinien sinnvoll erscheint (z.B. Systeme mit und ohne Reinraum-/ Lebensmitteleignung). Dennoch sollte hier auf Möglichkeiten zur Wiederverwendung geachtet werden. Beispielsweise können ggf. die selben Geometrien verwendet werden und nur die Materialwahl angepasst werden oder Verkleidungen (Kapselungen) und Beschichtungen vorgesehen werden. Für andere Anforderungsspreizungen zeigt sich, dass sie hohen Einfluss auf das Gesamtsystem haben und im Sinne eines modularen Baukastensystems auf einige feste Werte gesetzt werden können (z.B. Gelenke mit einem oder zwei Drehfreiheitsgraden, Arbeitsplattformen mit zwei bis sechs Anbindungsstellen für kinematische Ketten). Die erste in Abb. 5.13 angegebene Anforderungsspreizung, bezieht sich z.B. auf die geforderte Gelenksteifigkeit und ist dem *Package* Kräfte zugeordnet. Die geforderten Steifigkeiten liegen bei Werten $k > 10\,\mathrm{N}/\mu\mathrm{m}$ bei Montageaufgaben und $k > 5\,\mathrm{N}/\mu\mathrm{m}$ bei Handhabungsaufgaben. Je nach Randbedingungen des herstellenden Unternehmens können zwei Gelenkbaureihen (für Handhabung und für Montage) sinnvoll sein oder ein Gelenk mit der maximalen Steifigkeit wird in allen Robotern verbaut (hohe Anzahl von Gleichteilen).

Im Anforderungsmodell des Parallelroboters wurden die `trace`-Beziehungen wie in Abschnitt 4.1.6 beschrieben modelliert. Einige Anforderungen weisen in der durch das erarbeitete Makro erzeugten `trace`-Matrix (vgl. Abschnitt 4.2.6) eine

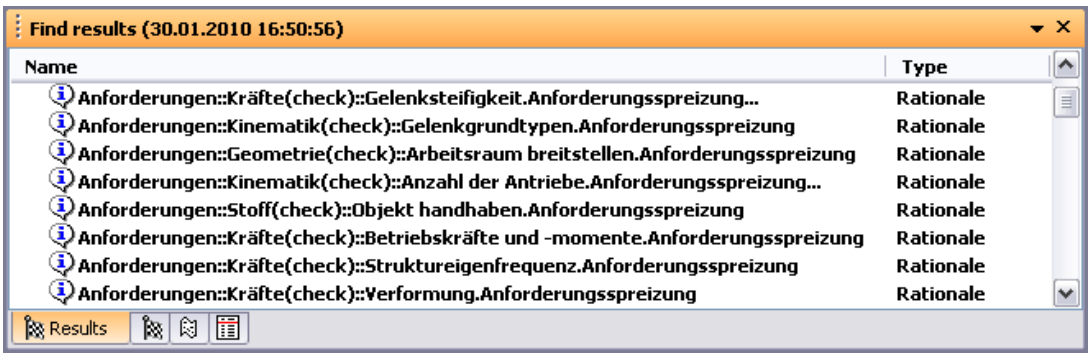


Abbildung 5.13: *Ergebnisse der Suche nach Anforderungsspreizungen im Modell.*

große Anzahl Beziehungen auf. Es wird untersucht, ob hier mehrere Anforderungen durch einen gemeinsamen Effekt in Beziehung stehen. Als einfaches Beispiel kann auf einem niedrigen Konkretisierungsgrad die tripolare Beziehung zwischen Beschleunigung, Geschwindigkeit und Arbeitsraumgröße durch eine mathematische Gleichung ausgedrückt werden. Es gilt unter der vereinfachten Annahme einer gleichförmig beschleunigten Bewegung:

$$v = v_0 + a \cdot t \tag{5.1}$$

$$s = v_0 \cdot t + \frac{1}{2} a \cdot t^2 \tag{5.2}$$

Durch Umformen von Gl. (5.1) nach t und einsetzen in Gl. (5.2) folgt:

$$s = \frac{v^2 - v_0^2}{2 \cdot a} \tag{5.3}$$

Wird eine konstante Beschleunigung am TCP ($a = a_{max}$) und ein Start aus der Ruhe (Anfangsgeschwindigkeit $v_0 = 0\,\mathrm{m/s}$) vorausgesetzt, ergibt sich durch Umformen der Gl. (5.3) nach einer zurückgelegten Strecke $s = l_{max}$ die Geschwindigkeit v_{max} zu:

$$v_{max} = \sqrt{2 \cdot a_{max} \cdot l_{max}} \tag{5.4}$$

Die maximale Länge der Trajektorie l_{max} (und damit die Bewegungsweite in die Raumrichtungen) kann in dieser Gleichung als Konstante aufgefasst werden, da sie durch den speziellen Anwendungsfall als Punktforderung vorgegeben wird. Der so gefundene quantitative Zusammenhang wird im Modell in einem Parametrikdiagramm festgehalten (s. Abb. 4.10). Die Beschleunigung wird in erster Linie durch die Wahl des Antriebs festgelegt (vgl. Abschnitt 5.2.7). Direktantriebe bieten hier die höchsten Beschleunigungen.

Die angebotene Leistung des Antriebs wird abhängig von der zu bewegenden Masse in Beschleunigung umgesetzt. Je geringer die bewegten Massen sind, desto höher ist die mögliche Beschleunigung am TCP. Durch eine gezielte Analyse der bereits im Modell vorhandenen Zusammenhänge zeigt sich, dass die bewegten Massen zum einen direkt durch die Nutzlast festgelegt und zum anderen durch die Massen der kinematischen Struktur beeinflusst werden. Die hängen wiederum von der Größe des Arbeitsraums und damit der Länge der Glieder ab. Außerdem müssen die kinematischen Ketten für höhere Nutzlast und bei größerer Länge höhere Flächenmomente bereitstellen, wodurch ebenfalls die Masse der Struktur erhöht wird.

Eine leichtbauoptimierte Roboterstruktur ist oftmals anfälliger bezüglich Schwingungsanregungen, die z.B. beim Abbremsen der Struktur auftreten können. Schwingt die Struktur beim Abbremsen so kann nicht die geforderte hohe Genauigkeit erreicht werden oder der Roboter muss vor jeder Manipulation das Abklingen der Schwingungen abwarten. Der Zeitgewinn einer hochdynamischen

Struktur wird dadurch aufgebraucht. Werden die Beziehungen bis in das Zielmodell weiterverfolgt zeigt sich, dass auch hier wieder ein Zielkonflikt zwischen hoher Dynamik und hoher Genauigkeit vorliegt. Im SFB 562 wurden adaptronische Komponenten eingeführt, um diese Schwingungen zu unterdrücken. Flächige Aktoren werden auf die kinematische Struktur aufgebracht, um durch gegenphasige Anregung die Schwingung schnell abklingen zu lassen [Kei08].

Die systematische Untersuchung der Support-Matrix (vgl. Abschnitt 4.2.6) zur Identifikation weiterer Zielkonflikte zeigt, dass der Großteil der Beziehungen unterstützenden Charakter aufweist. Ein bekannter Zielkonflikt auf Komponentenebene liegt zwischen einem geringen geforderten Spiel und einer geringen geforderten Reibung zwischen den beweglichen Elementen im Gelenk [Pav06]. Wird dieser Zielkonflikt auf Systemebene übertragen, so zeigt sich, dass ein kleines Gelenkspiel die Genauigkeit des Robotersytems unterstützt, während die geringe Reibung eine höhere Beschleunigung ermöglicht. Das erarbeitete Makro zur Zielkonfliktanalyse ist in der Lage auf Basis der bekannten Zusammenhänge diesen zusätzlichen Zusammenhang automatisch zu identifizieren (vgl. Abb. 5.14). Der Umstand (`instance`) dieses Zielkonflikts ist jedoch temporal, d.h. liegt nicht zeitgleich vor: Die hohen Genauigkeitsforderungen resultieren aus Anwendungsfällen der Montage, während die Beschleunigungsforderungen aus Anwendungsfällen des *Pick-and-Place* resultieren. Diese Zusammenhänge sind im Anforderungsdiagramm Abb. 5.15 visualisiert.

Für die Entwicklung von Robotern, die ausschließlich in einem der beiden Bereiche (Montage oder *pick-and-place*) genutzt werden sollen, erscheint die Lösung des Konflikts durch einen Kompromiss sinnvoll (vgl. Abschnitt 3.4.2). Für *Pick-and-Place*-Roboter können Gelenke für hohe Roboterbeschleunigungen optimiert werden, so dass sie noch eine akzeptable Genauigkeit ermöglichen. Soll der

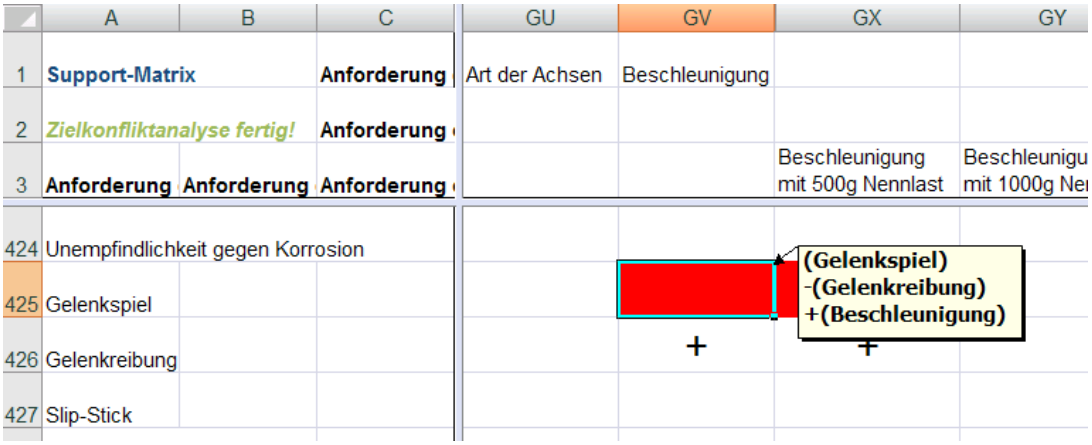


Abbildung 5.14: *Ausschnitt aus dem Excelarbeitsblatt „Support-Matrix“ nach erfolgter Zielkonfliktanalyse. Ein neu identifizierter, potenzieller Zielkonflikt ist markiert und der Weg im Kommentar angegeben.*

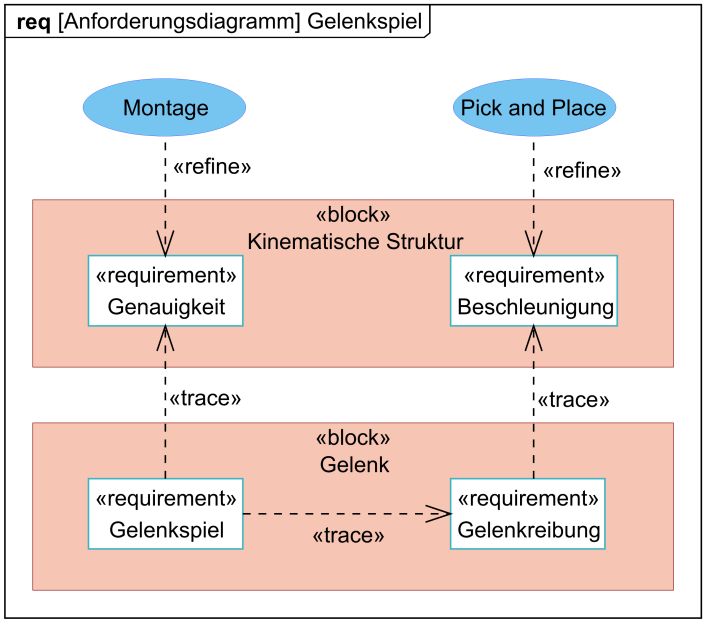


Abbildung 5.15: *Anforderungsdiagramm zur Verdeutlichung des temporalen Zielkonflikts zwischen Gelenkspiel und Gelenkreibung.*

Roboter für beide Anwendungsbereiche genutzt werden, führt ein Kompromiss zu einer für beide Bereiche unbefriedigenden Lösung. Zur Auflösung des Zielkonflikts können adaptronische Elemente in die Gelenke eingebracht werden [Pav08], so dass z.B. durch eine Verringerung des Spiels während des Abbremsvorgangs die Genauigkeit bei der Handhabung steigt. Durch eine Vergrößerung des Spiels werden bei der Beschleunigung die Reibungsverluste im Gelenk minimiert. Die konfliktären Eigenschaften werden hier durch Adaptronikeinsatz zeitlich voneinander getrennt.

Mit Hilfe der Support-Matrix werden zusätzlich indirekte Unterstützungen verdeutlicht. Beispielsweise ist bei einer hohen Gestellsteifigkeit eine hohe Beschleunigung am TCP möglich. Dieser Zusammenhang ist indirekt, d.h. bei gleicher Genauigkeit des Robotersystems ermöglicht eine höhere Gestellsteifigkeit eine höhere Beschleunigung. Nachdem diese Unterstützung in das Modell integriert wurde, zeigte sich nach erneutem Durchlauf der Zielkonfliktanalyse, dass hier ein Zielkonflikt entstehen kann: Die Steifigkeit wird z.B. durch ein größeres Flächenträgheitsmoment der Träger (bei Fachwerkbauweise) erhöht. Häufig erhöht sich damit auch der Flächeninhalt und somit die Masse. Eine höhere Beschleunigung am TCP zieht demnach häufig ein höheres Gesamtgewicht und damit höhere Herstell- und Logistikkosten nach sich. Es darf allerdings nicht der Fehler begangen werden diesen Zielkonflikt als unausweichlich anzusehen, da der Zusammenhang zwischen Steifigkeit und Masse eben nur indirekt besteht. Die Aussage „steife Bauteile sind immer schwer“ ist falsch (vgl. [Ste07a]).

Durch die Identifikation dieses möglichen Zielkonflikts sensibilisiert, kann die Gestaltung derart durchgeführt werden, dass der Konflikt nicht zum Tragen kommt: Beispielsweise kann ein Rundrohrprofil durch Vergrößerung des Durchmessers und gleichzeitiger Verringerung der Wandstärke bei gleichem Flächeninhalt deutlich höhere Flächenträgheitsmomente aufweisen als die Ausgangsgeometrie. Sind diese konstruktiven Freiheitsgrade jedoch bereits durch Restriktionen eingeschränkt, muss entweder ein Kompromiss geschlossen (z.B. Erhöhung der Steifigkeit bis zu einer gerade noch akzeptablen Massenzunahme) oder eine neuartige Lösung gefunden werden (z.B. neue Topologie oder anderes Material).

5.2.7 Entwicklung des Parallelrobotersystems

Wie in Abschnitt 4.1.7 beschrieben werden Anforderungen in den frühen Phasen zusammengestellt und auf unterschiedlichen Abstraktionsebenen Systemelementen zugeordnet. Dadurch kann das Produktmodell schrittweise konkretisiert werden. Auf jeder Ebene werden die Anforderungen mit den Parametern der betrachteten Module verglichen. Falls für ein Modul verschiedene Realisierungsmöglichkeiten für die darunterliegende Ebene existieren, kann der Entwickler diejenige Lösung auswählen, die am Besten zu dem betrachteten Anforderungskollektiv passt. Schließlich können konkrete bereits entwickelte Komponenten aus einer Datenbasis gewählt und zu der gewünschten Struktur zusammengesetzt werden [Fra05c]. Wird keine passende Komponente in der Datenbasis gefunden, so werden die vorliegenden Anforderungen genutzt, um eine möglichst gut passende Komponente zu finden und zu modifizieren oder eine komplett neue Lösung zu entwickeln.

Dieses Vorgehen verkürzt die Entwicklungs- und Herstellzeit und reduziert Komplexität und interne Varianten. Gleichzeitig wird die Qualität erhöht und das Entwicklungsrisiko gemindert, denn der Großteil der verwendeten Teile ist bereits erprobt und bewährt.

Das kinematische Verhalten von Parallelstrukturen wird hauptsächlich durch die Art und Anordnung der Gelenke innerhalb der kinematischen Ketten festgelegt. Daher beginnt die Festlegung der Module sinnvollerweise mit der Strukturssynthese. Für ein lebenslauforientiertes Baukastensystem werden Modelle unterschiedlicher Abstraktionsgrade und Modellierungstiefe benötigt [Fra05d] [Ste06b]. Ausgehend von einem abstrakten objektorientierten Modell werden das kinematische, das dynamische und das geometrische Modell aufgebaut.

Für ein Baukastensystem werden, Schritt 4 aus Abb. 4.27 folgend, die Systemkomponenten etabliert. Das entwickelte abstrakte Strukurschema (Abb. 5.16) beinhaltet zehn Module, die als Blöcke in SysML-Notation dargestellt sind. Jeder Block ist eine Art Container, der charakteristische Parameter des beschriebenen Moduls enthält, z.B. Länge eines Stabes (vgl. [Wah03, S. 64]). Eine Anzahl von definierten Schnittstellen ist zwischen den Modulen jeweils als durchgezogene Linie dargestellt. Durch arrangieren der Module entlang der Linien werden Strukturen

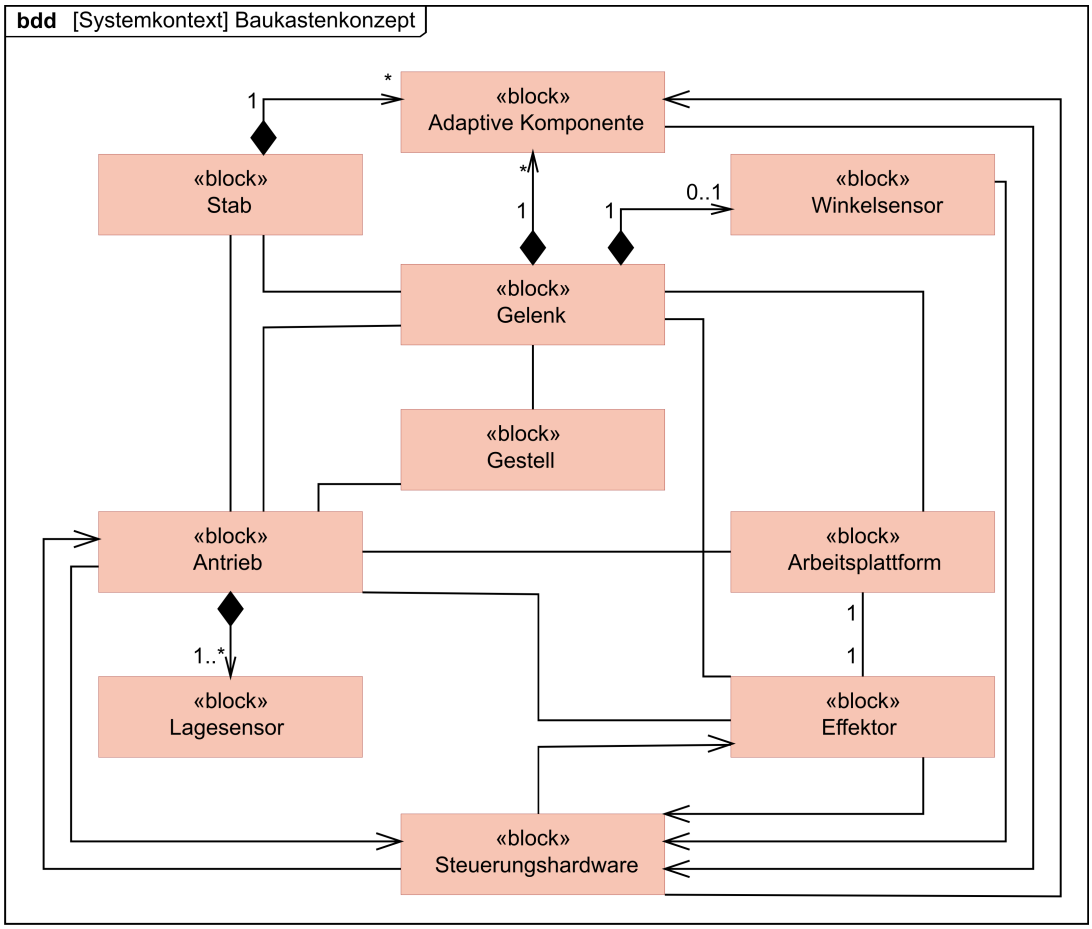


Abbildung 5.16: *Abstraktes Strukturschema des Baukastenkonzepts (— Schnittstelle, → Datenpfad).*

aufgebaut. Dabei können Module mehrfach verwendet, d.h. die Linien in beide Richtungen mehrmals verfolgt werden (vgl. [Ste06b]). Nicht alle Module müssen genutzt werden, sind also Kann-Bausteine.

Die gezeigten Blöcke werden weiter in Submodule zerlegt [Ste06b]. Die hierfür notwendigen Beziehungen und Gestaltungsregeln werden im objektorientierten Modell dargestellt. So können konkrete Roboterstrukturen vom Groben zum Detaillierten aufgebaut werden. Je detaillierter das Produktmodell wird, desto aufwendiger und unübersichtlicher wird allerdings ein abstraktes Modell. Daher muss zum rechten Zeitpunkt auf eine weiterführende Modellierungsart (z.B. CAD) umgeschwenkt werden. Das abstrakte Modell bleibt jedoch in der übergeordneten Ebene erhalten.

Abb. 5.17 zeigt einen Ausschnitt aus einem Beziehungssystem als ein Beispiel für die Überlegungen der Baukastenentwicklung innerhalb dieses Konzepts. Es ist zu sehen, dass der aus der Lebensmittelindustrie (genauer Gebäckverpackung)

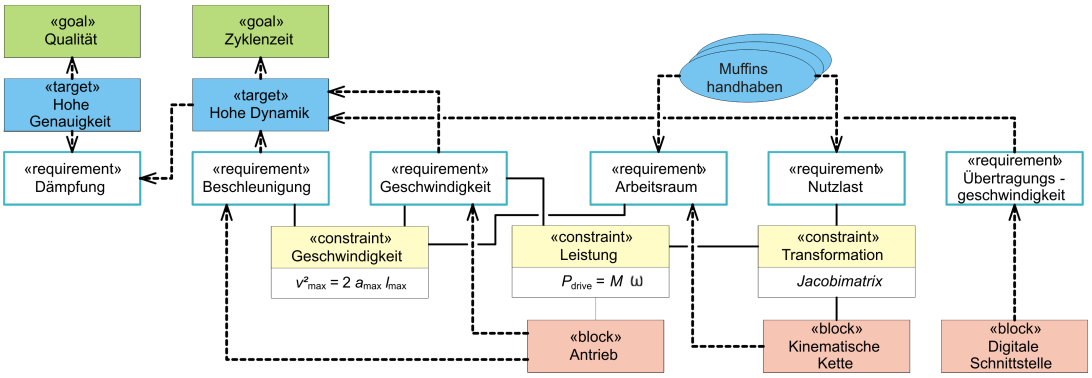


Abbildung 5.17: *Ausschnitt aus einem Diagramm zur Darstellung der Intermodellbeziehungen als Grundlage für eine Einflussanalyse.*

stammende Anwendungsfall *Muffins handhaben* die Anforderungen *Arbeitsraumdimension* und *Nutzlast* verfeinert. Durch Objekt (Muffin) und Greifer ergibt sich eine geforderte Nutzlast der Struktur von etwa 1 kg. Die Arbeitsraumdimension hängt von der Realisierung der Produktumgebung ab, d.h. der Bereich der Muffinanlieferung, der Bereich der Ablage und der geforderten Höhe zum Anheben der Muffins. Die Muffins müssen mindestens um ihre eigene Höhe gehoben werden, damit sie nicht mit den noch auf dem Band befindlichen Muffins kollidieren. Aus dem Bild ist zudem ersichtlich, dass die Geometrie und Größe des Arbeitsraums durch Art und Anordnung der kinematischen Ketten bestimmt werden.

Diese Überlegungen führen zu einem System von verschiedenen Baukästen auf unterschiedlichen Ebenen. Durch Basisanforderungen (z.B. Freiheitsgrad, Arbeitsraumgeometrie) wird die kinematische Struktur festgelegt (z.B. Delta, Hexa). Dabei werden die jeweils passenden Komponenten (z.B. Stab, Gestell) ausgewählt. Komponenten können in rechnerunterstützten Konstruktionskatalogen (vgl. [Fra01], [Fra04]) abgelegt werden. Diese stehen dann als übersichtliche Lösungssammlung zur Verfügung (vgl. Abb. 5.18).

Auf der nächsten Ebene sind die Komponenten selbst als Baukasten dargestellt. Beispielsweise kann ein kinematisch erforderliches Kugelgelenk durch eine Hintereinanderschaltung eines Kardan- und eines Drehgelenks substituiert werden (vgl. Hexa des SFB 562 [Otr05]). Stäbe bestehen aus einem Grundkörper und Verbindungselementen an jeder Seite. Der Grundkörper wird mit verschiedenen Profilen und Materialien, sowie mit und ohne Adaptronik ausgeführt. Die Verbindungselemente stellen unterschiedliche Schnittstellenarten bereit (z.B. Außengewinde, Innengewinde oder Klebung in verschiedenen Größen). Außerdem ermöglichen die Verbindungselemente unterschiedliche Orientierungen der Anbindungen, z.B. Gelenkanschluß in Stabachse (vgl. Hexa) und senkrecht dazu (vgl. Fünfgelenk).

Für viele Strukturen besteht die Möglichkeit eine Variante der Kurbel (beim Hexa: Das Glied zwischen Antrieb und Kardangelenk) zu verwenden, die mehrere

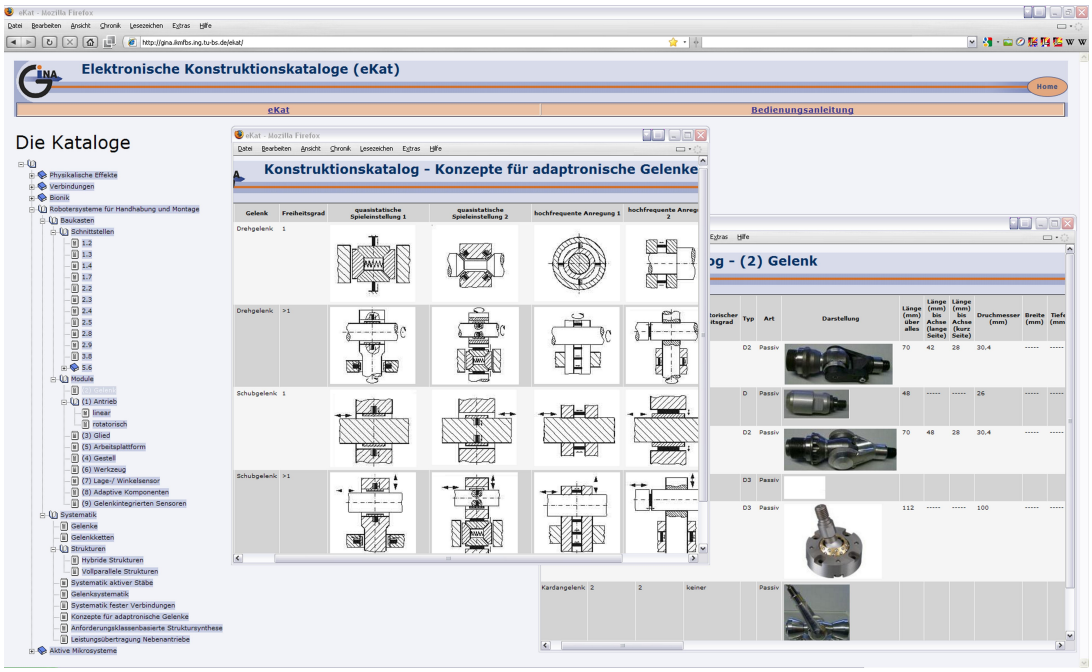


Abbildung 5.18: *Übersicht der für die Parallelroboterentwicklung zusammengestellten rechnerunterstützten Konstruktionskataloge und zwei Beispielkataloge für die Auswahl passiver Gelenke (vgl. [Otr05]) bzw. die Entwicklung adaptronischer Gelenke (vgl. [Pav06]).*

Anschlußstellen für Gelenke besitzt, d.h. die Gelenke können an mehreren festgelegten Stellen entlang der Kurbellänge montiert werden. Diese Möglichkeit spart eine erhebliche Anzahl von Varianten und erlaubt sogar eine einfache statische Rekonfiguration der Struktur (vgl. auch Abschnitt 5.3). Die Einflussanalyse (vgl. Schritt 5 in Abb. 4.27: Verfolgung relevanter Beziehungen im Diagramm durch Verwendung der `populate`-Funktion, s. Abschnitt 4.2.7) zeigt allerdings, dass die Kurbel für einen Großteil der Varianten überdimensioniert und deshalb schwerer und träger wäre als notwendig. Darüberhinaus würde sich der regelungstechnische Aufwand zur Vermeidung von Strukturkollisionen erhöhen. Die Ziele *hohe Dynamik* und *geringer Energieverbrauch* während des Betriebes wären ebenfalls beeinträchtigt.

Einen besonders hohen Anteil an den Selbstkosten des Herstellers hat das Robotergestell. Eigene Untersuchungen haben ergeben, dass etwa ein Drittel der Selbstkosten der mechanischen Komponenten auf das Gestell entfallen. Grundlage der Analyse waren die im Rahmen des SFB entwickelten Versuchsmuster Hexa und Fünfgelenk. Für die Gegenüberstellung wurden die Material- und Fertigungslohnkosten bzw. Kosten für Kaufteile detailliert betrachtet. Zwar wäre diese Analyse für eine Preisfindung bei Serienproduktion noch zu ungenau, der hohe Kostenanteil der Gestelle ist dennoch eindeutig belegbar.

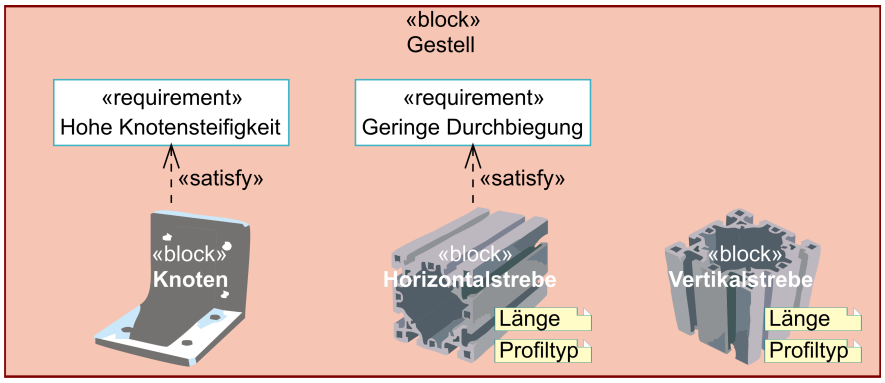


Abbildung 5.19: *Konzept zur Realisierung von flexiblen, rekonfigurationsfähigen Gestellen: Bausteine und erfüllte Anforderungen.*

Gestelle werden häufig individuell für jeden einzelnen Roboter und dessen spezielle Umgebung entwickelt und gefertigt, da sie einen großen Einfluss auf die Systemsteifigkeit, Anordnung der Antriebe, Arbeits-/ Aufstellraum und Zugänglichkeit haben (vgl. Abschnitt 5.2.6).

Für die Entwicklung des Gestellbaukastens wurden folgende Punkte in Betracht gezogen:

- Der Baukasten muss als offenes System flexibel hinsichtlich der verwendeten Strukturen und der speziellen Umgebung am Aufstellort sein.
- Das Gestell muss die statische Rekonfiguration, d.h. Änderung der Art, Position und Orientierung der Antriebe, erlauben (vgl. Abschnitt 5.3).

Nach Betrachtung der funktionalen Aspekte wurde ein Baukastensystem gewählt, welches auf standardisierten und gut erhältlichen Strangpressprofilen beruht (s. Abb. 5.19) [Bec06]. Verbindungen zwischen den Elementen sind flexibel einzustellen und—da lösbar—zur Rekonfiguration geeignet. Zwar sind die Halbzeuge hier wesentlich teurer als bei einer Schweißkonstruktion, die erreichte Flexibilität, der kostengünstigere Aufbau und Einsparungen durch einen einfacheren Transport bei Vor-Ort-Montage wiegen diesen Nachteil jedoch deutlich auf.

Weiterführende Analysen ergaben, dass die lösbaren Verbindungen zwischen den Elementen gerade bei der hochdynamischen Belastung der Struktur wesentliche Schwachstellen bilden [Sur09]. Für eine Umsetzung des Baukastenkonzepts müssen die Verbindungselemente hinsichtlich der speziellen Anforderungen sorgfältig gestaltet werden.

5.2.8 Testen des Parallelroboters

Qualitative Regeln und Restriktionen sind allein nicht ausreichend, um eine gute Lösung für eine spezielle Roboteraufgabe zu finden [Mer05]. Simulationen sind

notwendig, um entwickelte Lösungen zu überprüfen und mögliche—aber nicht zufriedenstellende—Varianten auszuschließen. Die üblichen (z.B. CAD, FEM, MKS) und die speziell angepassten rechnerunterstützten Werkzeuge (z.B. MAPS zur kinematischen Analyse) müssen dafür sinnvoll ineinander greifen. Dieses kann durch eine Kopplung oder Integration der Systeme erfolgen [Ada92]. Eine integrierte Entwicklungsumgebung muss umfangreiche Datenbanken verknüpfen und standardisierte systemübergreifende Komponenten nutzen [Goh98]. Das System soll den Anwender durch den gesamten Produktentwicklungsprozess leiten und ihn beim Datenaustausch unterstützen. Zudem soll es helfen, unnötige Iterationen und wiederholten Modellaufbau nach geringfügigen Änderungen zu vermeiden. Darüberhinaus sollen verschiedene Entwickler gleichzeitig an dem gleichen Projekt mit aktuellen Datensätzen arbeiten können. Dazu muss eine Entwicklungsumgebung eine konsistente Datenbasis zur Verfügung stellen.

Die im Rahmen des SFB 562 entwickelte Umgebung stellt einem Projektverantwortlichen einen konkreten Leitfaden zur Unterstützung seiner Koordinierungsarbeit bereit [Fra05d] [Fra05c] [Wah06, S. 121]. Dabei werden wie in Abb. 5.20 gezeigt, zuerst die Basisanforderungen mit einem einfachen Anforderungswerkzeug geklärt. Darauf aufbauend erfolgt eine systematische Strukturssynthese, deren Ergebnis ein Satz grundlegender Strukturparameter ist. Diese werden im xml-Format strukturiert abgelegt. Aus dieser Datenbasis werden Modelle für

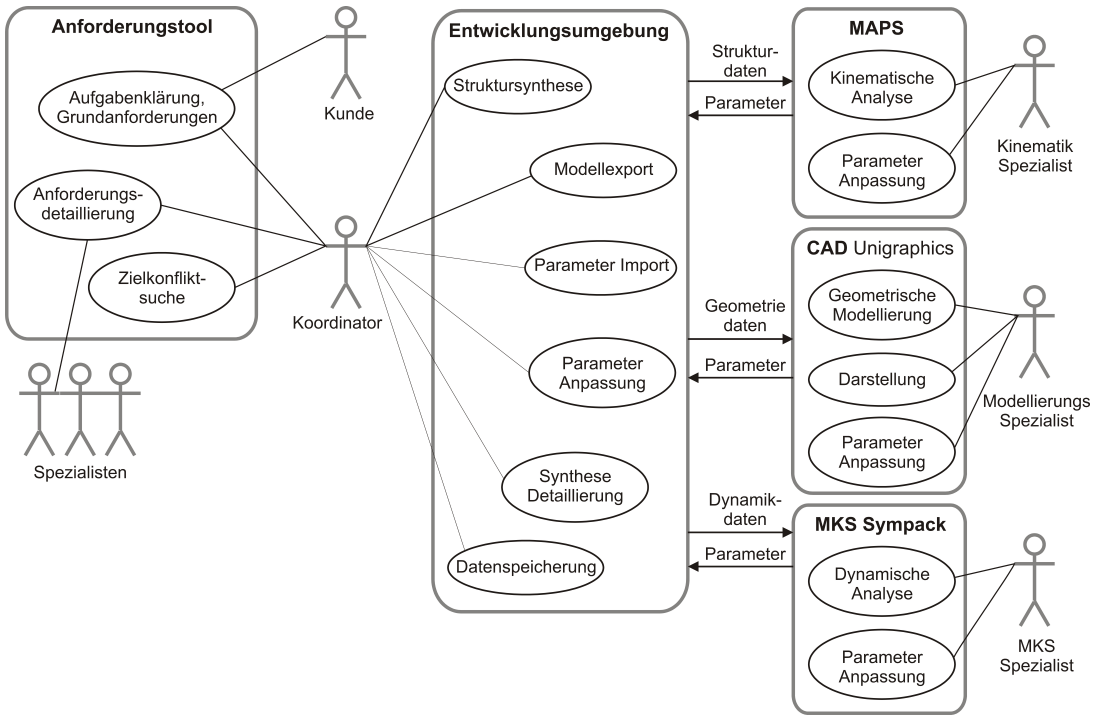


Abbildung 5.20: *Anwendungsfalldiagramm der Funktionalitäten der im SFB 562 erarbeiteten Entwicklungsumgebung, nach [Wah06, S. 121].*

die angebundenen Softwarewerkzeuge zur kinematischen Analyse, geometrischen Modellierung und dynamischen Analyse semiautomatisch erzeugt. Auf Basis prototypischer CAD-Teile werden physikalische Eigenschaften ermittelt und in den anderen Werkzeugen bereitgestellt (z.B. Massen, Trägheitsmomente). Außerdem werden aus dem Katalogsystem vorhandene Bauteile ausgewählt und damit die prototypischen Modelle ersetzt. Die Konsistenz der Daten wird durch eine Regelverarbeitung sichergestellt. Falls ein Parameter des reduzierten Parametersatzes in einem Werkzeug verändert wird, errechnet die Entwicklungsumgebung beim Datenimport die ebenfalls zu ändernden abhängigen Parameter des vollständigen Parametersatzes.

Die beschriebene Entwicklungsumgebung ist in der Lage, konsistente Datensätze mit verschiedenen Simulationswerkzeugen auszutauschen. Aber gegen welche Anforderungen kann in welchen Werkzeugen am sinnvollsten getestet werden und wie sehen die zugehörigen Testfälle aus?

In Abschnitt 4.1.8 wurde eine Systematisierung von Testfällen vorgestellt und in Abb. 4.14 abgebildet. Zuerst werden aus dem im Modell vorhandenen Anforderungen Testkriterien formuliert. Anforderungen und Testkriterien werden solange überarbeitet bis ein konsistenter Katalog vorliegt und alle Anforderungen testbar sind (vgl. Abschnitt 4.2.8). Die Testkriterien werden jeweils den verschieden Testfallklassen und schließlich konkreten Testfällen zugeordnet. Diesen werden die konkreten Testwerkzeuge und Testpersonen zugeordnet. Abb. 5.21 stellt diese Zusammenhänge für den Testfall „*Arbeitsraumanalyse*“ grafisch dar.

Im Rahmen der oben beschriebenen Entwicklungsumgebung können die Klassen von Testfällen (TC - *testcase*) überprüft werden, die sich auf dem Aggregationslevel *Maschine* befinden:

- TC 7: Testet die Funktion auf virtueller Ebene. Beispiel:
  Vorbedingung: Das CAD-Modell der kinematischen Struktur liegt vor und die wesentlichen Randbedingungen (z.B. zugelassene Freiheitsgrade der Gelenke) sind im Modell festgelegt. Die Eingangsgrößen (Drehwinkel der Antriebe) sind durch Parameter steuerbar. Eine Trajektorie ist im Modell enthalten und die Arbeitsplattform ist daran beweglich angebunden. Die Arbeitsplattform befindet sich am Startpunkt der Testtrajektorie.
  Durchführung: Die Struktur wird durch „ziehen“ an der Arbeitsplattform entlang der Trajektorie bewegt. Es wird überprüft, ob die kinematische Struktur die Bewegung ohne Strukturkollisionen durchführen kann und es nicht zum „Verklemmen“ z.B. durch Gelenkwinkelbeschränkungen kommt.
  Nachbedingung: Die Arbeitsplattform befindet sich am Endpunkt der Testtrajektorie. Es wurden keine Kollisionen oder „Verklemmungen“ festgestellt. Die Funktion ist erfüllbar.
  Einschränkungen: Dieser Testfall berücksichtigt nicht das Vorhandensein etwaiger Singularitäten im Arbeitsraum. (Singuläre Bereiche müssen durch eine Analyse der Jacobi-Matrix identifiziert werden, z.B. [Fri01] [Sch09].)

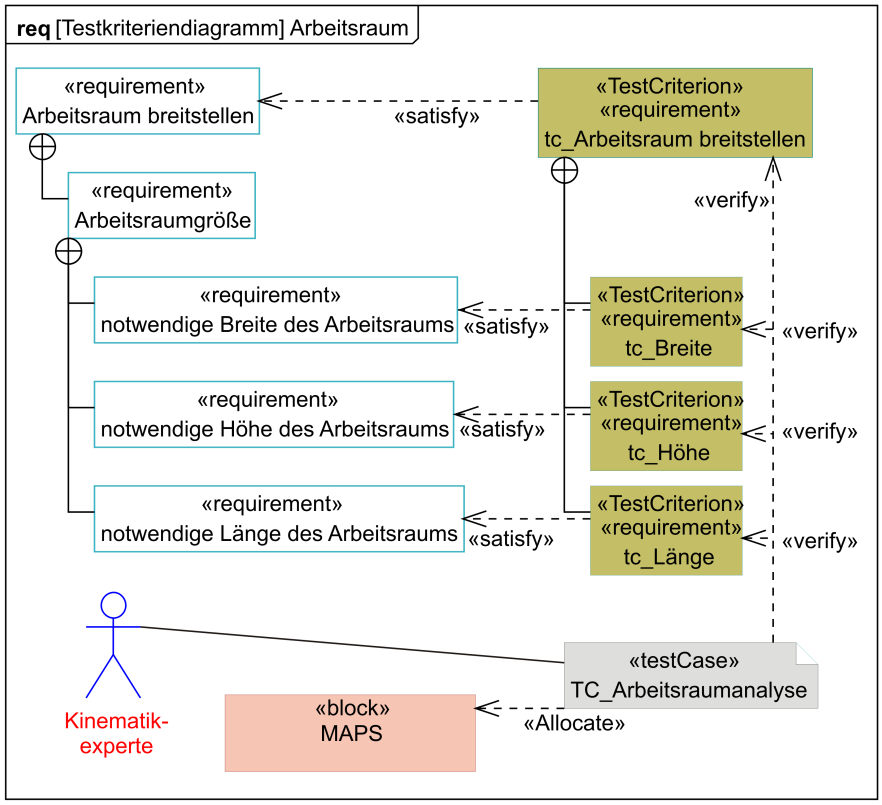


Abbildung 5.21: *Einordnung des Testfalls zur Arbeitsraumanalyse zu Anforderungen, Testkriterien, Testfallklassen, Testwerkzeug und Testpersonen.*

- TC 19: Testet die Struktur auf virtueller Ebene. Beispiel (s. auch Abb. 5.21):
  Vorbedingung: Ein MAPS-Modell der kinematischen Struktur liegt vor. Die wesentlichen Randbedingungen (z.B. Winkeleinschränkungen der Gelenke) sind im Modell festgelegt.
  Durchführung: Die Berechnung der Eigenschaften des Arbeitsraums wird gestartet. Es wird überprüft, ob in dieser Konfiguration die gewünschte Größe des Arbeitsraums erreicht wird und ob dieser Arbeitsraum frei von Singularitäten ist.
  Nachbedingung: Es liegen Diagramme vor, welche die Größe des Arbeitsraums beschreiben, sowie die Lage etwaiger Singularitäten angeben. Der nutzbare Arbeitsraum ist groß genug und frei von Singularitäten. Das Strukturkriterium ist erfüllbar.
  Einschränkungen: Dieser Testfall berücksichtigt nicht die Gestaltung der Struktur. Strukturkollisionen können nicht erkannt werden.

- TC 31: Testet das Verhalten auf virtueller Ebene. Beispiel:
  Vorbedingung: Ein MKS-Modell der kinematischen Struktur liegt vor. Die wesentlichen Randbedingungen (z.B. Massen, Trägheitsmomente) sind im

Modell festgelegt. Eine Testtrajektorie ist im Modell enthalten.
Durchführung: Die Simulation wird gestartet. Es wird überprüft, ob die gewünschten Beschleunigungen erreicht werden und die dynamischen Antriebskräfte die Grenzwerte nicht überschreiten.
Nachbedingung: Es liegen Diagramme vor, die die erzielten Beschleunigungen, Geschwindigkeiten und Kräfte an den Antrieben bzw. am Endeffektor darstellen. Die erreichten Beschleunigungen erfüllen das Testkriterium und die Grenzwerte sind eingehalten.
Einschränkungen: Dieser Testfall berücksichtigt die Gestaltung der Struktur nur näherungsweise. Die Komponenten werden als starre Körper angenommen und die Massenverteilung angenähert.

Eine Verbesserung der Ergebnisse kann, z.B. durch den Übergang auf Testfälle der realen Ebene gelingen. Beispielsweise kann für TC 11 aus den CAD-Daten ein (ggf. verkleinerter) Prototyp im Rapid Prototyping (RP)-Verfahren hergestellt werden. Jetzt ist eine Manipulation von Hand möglich, die eine direktere Interaktion und damit intuitive Funktionsüberprüfung ermöglicht.

Die Testkriterien-Testfälle-Matrix (vgl. Abschnitt 4.2.5) zeigt, dass hier insbesondere auf Ebene der Anlage reale Testfälle notwendig sind. Eine Abbildung der realen Gegebenheiten in einem virtuellen Testmodell ist derzeit noch sehr aufwändig. Auf Maschinenebene sind dagegen eine große Anzahl von Testkriterien virtuell überprüfbar. Für eine genauere Aussage fehlt hier jedoch noch eine detaillierte Ausarbeitung aller Testfälle und Zuweisung der Testkriterien.

5.3 Rekonfigurationsparameter identifizieren

Die immer breiter gefächerten Produktpaletten und immer schnelleren Modellwechsel erzeugen einen Bedarf an immer flexibleren Produktionsmitteln. Im Bereich der Parallelroboter lässt sich ein Trend, zu während der Nutzungsphase rekonfigurierbaren Robotern, ausmachen [Ber00]. In diesem Abschnitt wird gezeigt, wie das nach dem entwickelten Konzept erarbeitete Modell genutzt werden kann, um Parameter für eine sinnvolle Rekonfiguration zu identifizieren.

Nach *Setchi* kann unter Rekonfigurierbarkeit allgemein „*the ability to repeatedly change and rearrange the components of a system in a cost-effective way*“ verstanden werden [Set04]. Für Parallelroboter lassen sich drei Arten von Rekonfiguration unterscheiden [Kre06b]:

- Statische Rekonfiguration: Austausch und/ oder Demontage und Neuanordnung von Komponenten oder Baugruppen.

- Dynamische Rekonfiguration (Typ 1): Die Struktur erreicht durch Durchfahren von Singularitäten vom Typ 1 neue Arbeitskonfigurationen, die einen anderen Arbeitsraum bereitstellen.

- Dynamische Rekonfiguration (Typ 2): Spezielle Komponenten werden genutzt, um Eigenschaften im Betrieb quasistatisch zu verändern (z.B. ausfahrbare Stäbe, blockierbare Gelenke).

Aus der Literatur sind verschiedene Ansätze zur Entwicklung rekonfigurierbarer Mechanismen bekannt [Kor99] [Pri00] [Jov02] [Wur02] [Yoo02] [Kre06b] [Bud09]. Diese Ansätze berücksichtigen die entwicklungsmethodischen Aspekte nur sehr allgemein oder sie sind sehr spezifisch auf die Rekonfiguration einer bestimmten Struktur ausgerichtet. Ein Ansatz, der ausgehend von Kundenbedürfnissen die sinnvoll anzupassenden Parameter identifiziert, fehlt bisher.

Rekonfiguration sollte ein bestimmtes Ziel auf Grundlage einer strategischen Entscheidung verfolgen. Der Hersteller des Roboters muss sich bei der Planung über die sinnvollen Rekonfigurationsmöglichkeiten im Lebenslauf des Roboters bei einem bestimmten Kunden im Klaren sein. Eine systematische Analyse der Ziele, sowie eine Identifikation von zweckmäßigen Szenarien und relevanten Parametern und deren Einflüsse auf das Gesamtsystem müssen durchgeführt werden, damit ein Produkt erfolgreich am Markt sein kann. Der hier gezeigte Ansatz hilft systematisch jene Anforderungen zu identifizieren, die—als eine Repräsentation der Kundenwünsche—sich für verschiedene Anwendungsfälle voneinander unterscheiden (Anforderungsspreizung, vgl. Abschnitt 2.3.1). Diese Anforderungen führen schließlich zu den rekonfigurationsrelevanten Parametern und damit zu zweckmäßigen Konzepten. Der Ablauf folgt dem Bereits in Abschnitt 4.3 angegebenen Vorgehen zur Baukastenentwicklung:

1. Strukturierung der Anwendungsfälle und Angabe der Beziehungen zu Anforderungen.

2. Aufbau des hierarchischen Zielsystems und Identifikation der essentiellen Anforderungen für Konzipierung und Gestaltung.

3. Analyse, ob die Anforderungsspreizungen durch Rekonfiguration umzusetzen sind.

4. Angabe der Beziehungen von Anforderungen zu Systemkomponenten und deren Parametern.

5. Analyse, wie diese Parameter zur Rekonfiguration nutzbar sind und welchen Einfluss ihre Änderung auf das Gesamtsystem hat.

Innerhalb des ersten Schrittes im Ablauf wird ein breites Feld von möglichen Anwendungsfällen in der Nutzungsphase betrachtet (z.B. Handhabung von Muffins, Beschickung von Einrichtungen zum Sandstrahlen, Bestücken von Leiterplatten). Diese Anwendungsfälle werden auf der einen Seite hinsichtlich der durchzuführenden Aufgaben (z.B. Pick-and-Place, Beschickung, hochpräzise Montage) und

auf der anderen Seite nach Branchen (z.B. Lebensmitteltechnologe, Fertigungstechnik, Informations- und Kommunikationstechnik) strukturiert. Darüber hinaus wird der Anwendungsfall Rekonfiguration in die Betrachtung einbezogen.

Nun werden `refine`-Beziehungen zwischen Anwendungsfällen und Anforderungen gesetzt. Beispielsweise verfeinert der Anwendungsfall *Handhabung von Muffins* die Anforderung *Arbeitsraumbreite* auf den Wert 500 mm. Dafür wird eine Unteranforderung gebildet, die genau diesen Wert erhält. Der Anwendungfall *Bestücken von Leiterplatten* verfeinert die gleiche Anforderung auf den Wert 250 mm. Es entsteht hier demnach eine Anforderungsspreizung (s. Abb. 5.22).

Der zweite Schritt im Ablaufplan dient dem Aufbau des hierarchischen Zielsystems. Ausgehend von den zuvor identifizierten Anforderungen werden Beziehungen zu den übergeordneten Zielen und den Objekten der Systemidee gebildet. Außerdem werden die sich aus den betrachteten Anforderungen ableitenden Anforderungen auf ihre Rekonfigurationsrelevanz hin überprüft und ggf. berücksichtigt. Zur Darstellung wird ein Anforderungsdiagramm verwendet. Hier können die Anforderungen per *drag-and-drop* aus dem *package-browser* eingefügt werden. Die durch das Programm bereitgestellte Funktion `populate` ermöglicht es auf Mausklick die mit dem betrachteten Objekt in Beziehung stehenden Objekte ebenfalls im Anforderungsdiagramm abzubilden.

Zunächst werden ausschließlich die konzeptrelevanten Anforderungen analysiert. In späteren Entwicklungsschritten (nachdem mögliche Konzepte eingegrenzt wurden) werden dann auch die gestaltungsrelevanten Anforderungen berücksichtigt. Beispielsweise haben die verschiedenen zu brücksichtigenden Nutzlasten und Prozesskräfte Einfluss auf die relevante Anforderung *Betriebskraft*. Die auf Systemebene geforderte Genauigkeit wird auf Subsystemebene unter anderem durch die maximal zulässige elastische Verformung der kinematischen Struktur

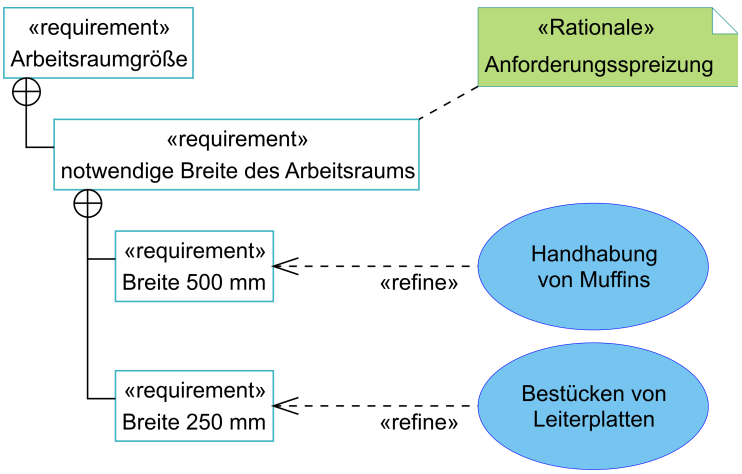


Abbildung 5.22: *Anwendungsfalldiagramm zur Darstellung der strukturierten Anwendungsfälle und deren Einfluss auf die Werte von Anforderungen.*

beeinflusst. Der Zusammenhang zwischen beiden Anforderungen wird durch die strukturelle Steifigkeit der kinematischen Struktur als essentielle Anforderung hergestellt (vgl. Abb. 5.23).

Im dritten Schritt werden nun die rekonfigurationsrelevanten Anforderungsspreizungen untersucht. Nicht zwischen jedem Anwendungsfall ist ein Umbau zweckmäßig. Beispielsweise kann ein *pick-and-place* Roboter der Lebensmittelindustrie zuerst Croissants und später Muffins handhaben oder er kann von einer *pick-and-place*-Aufgabe zu einer Verpackungsaufgabe für das gleiche Produkt im gleichen Unternehmen umgebaut werden. Im Gegensatz dazu ist es kaum denkbar, dass ein Roboter der chemischen Industrie an ein Unternehmen der Lebensmittelindustrie verkauft, rekonfiguriert und dort wiederverwendet wird. Der Roboter könnte zuvor mit Chemikalien in Kontakt gekommen sein, die unter ungünstigen Umständen an die Lebensmittel abgegeben werden könnten.

Die Änderung einiger Anforderungen kann weitere Anforderungen negativ beeinflussen, so dass auch hier eine Rekonfiguration nicht sinnvoll ist. Eine übersichtliche Überprüfung der Beeinflussung ist durch die in Abschnitt 4.2.6 beschriebene *Support*-Matrix möglich. Im konkreten Fall kann außerdem ein Diagramm erstellt werden, in dem sämtliche relevanten Beziehungen abgebildet und nachverfolgt werden. Beispielsweise liegt für den Freiheitsgrad am Effektor eine große Anforderungsspreizung vor. Soll bei einer vollparallelen Struktur ein zusätzlicher Freiheitsgrad durch eine zusätzliche kinematische Kette erreicht werden, so muss auch eine neue Steuerung entwickelt und implementiert werden. Die Arbeitsraumeigenschaften sind vollständig verschieden von der ursprünglichen Version. Die Rekonfiguration wird in diesem Fall durch den Aufwand der Steuerungsanpassung sehr teuer. Eine Alternative dazu ist eine hybride Lösung. Der zusätzliche Freiheitsgrad wird durch einen seriell angebrachten Antrieb realisiert. Die Anpassungen an der Steuerung sind minimal. Allerdings wird die Nutzlast und die Dynamik dadurch herabgesetzt.

Die systematische Analyse des Modells ergab den folgenden minimalen Satz von essentiellen rekonfigurationsrelevanten Anforderungen:

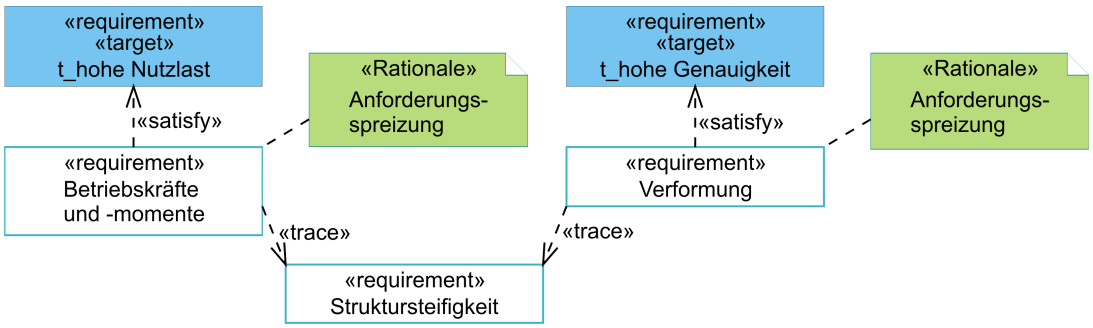


Abbildung 5.23: *Hierarchisches Zielsystem zur Identifikation der rekonfigurationsrelevanten Anforderungen.*

- geforderte Objekthandhabung erlauben,
- großen Arbeitsraum bereitstellen,
- hohe Betriebskräfte zulassen,
- notwendigen Freiheitsgrad bereitstellen,
- hohe Genauigkeit bereitstellen,
- hohe Leistungsfähigkeit erreichen und
- geringe Lebenslaufkosten erreichen.

Diese allgemein gehaltenen Anforderungen werden durch Anwendungsfälle weiter detailliert. Beispielsweise fallen auf dieser Ebene unter Freiheitsgrad nicht nur die Anzahl, sondern auch die Art der Bewegungsfreiheiten (z.B. Rotation um Z) und die jeweilige Orientierbarkeit (z.B. $\pm 25^o$). Die Anforderungsspreizungen liegen z.B. für die geforderte Wiederholgenauigkeit zwischen $\pm 1\,\mu$m bei Kleinteilmontage und $\pm 0,1$ mm bei *Pick-and-Place*-Aufgaben.

Innerhalb des vierten Schritts werden Anforderungen den Systemkomponenten zugeordnet, die „verantwortlich“ für deren Erfüllung sind. Die Systemkomponenten werden im Partialmodell Systemkontext dargestellt und sind Hard- und Softwarekomponenten. Diese Komponenten werden fallweise als Subsysteme weiter zerlegt und sind durch Schnittstellen mit weiteren Systemkomponenten verbunden. Beispielsweise wird, wie in Abb. 5.24 dargestellt, die Anforderung *Arbeitsraumgröße* durch die kinematische Struktur erfüllt, d.h. genauer durch die kinematischen Ketten und noch genauer durch die Länge der Glieder und Winkelbereiche der Gelenke. Zur Wahrung der Übersichtlichkeit sind im Diagramm nur die `allocate`-Beziehungen auf die letzte Ebene (Produktmerkmale: *Attributs*) dargestellt.

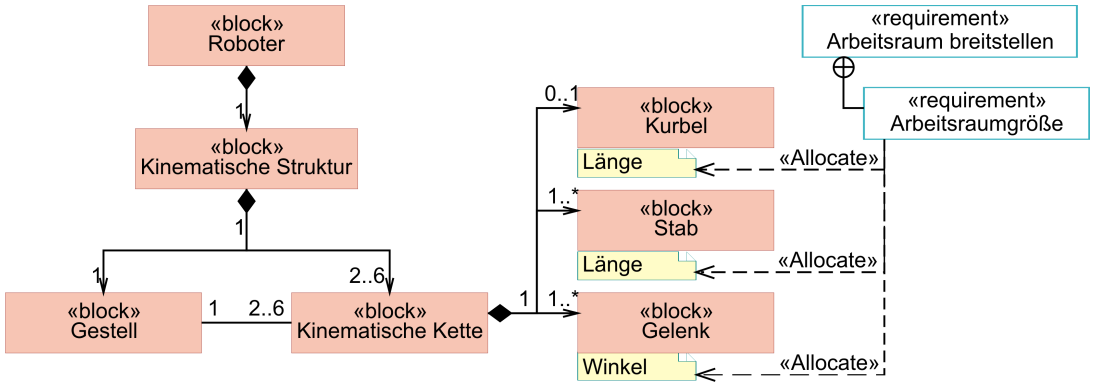


Abbildung 5.24: *Diagramm zur Darstellung der Beziehungen zwischen rekonfigurationsrelevanten Anforderungen und Systemkomponenten.*

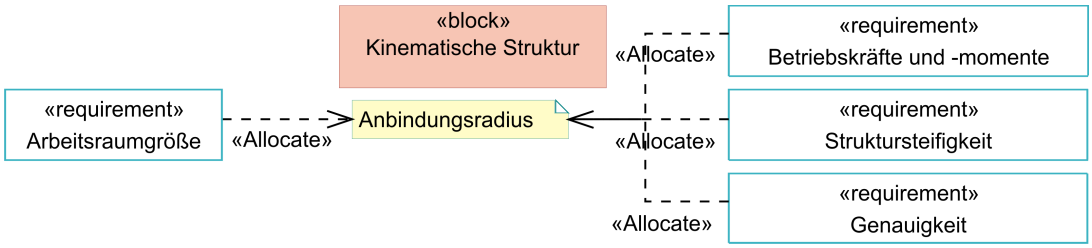


Abbildung 5.25: *Analyse des Einflusses eines möglichen Rekonfigurationsparameters „Anbindungsradius“ auf das Gesamtsystem.*

Im letzten Schritt werden die Beziehungen von den Anforderungen zu den Parametern verfolgt und mindestens qualitativ—besser quantitativ—beschrieben. Auf dieser Basis werden Rekonfigurationsszenarien entwickelt und analysiert. Insbesondere ist es notwendig zu untersuchen, ob die zur Rekonfiguration vorgesehenen Parameter weitere—bisher nicht berücksichtigte—Anforderungen beeinflussen und dadurch möglicherweise Zielkonflikte hervorrufen. Beispielsweise ist in Abb. 5.25 zu sehen, dass die Arbeitsraumgröße des Roboters durch eine Veränderung des Anbindungsradius der kinematischen Struktur zu erreichen ist. Im Modell sind bereits Beziehungen vom *Attribute* Anbindungsradius zu weiteren Anforderungen dokumentiert. Das bedeutet, dass eine Anpassung des Arbeitsraums durch Veränderung des Anbindungsradius ebenfalls zu Veränderungen bei den ertragbaren Betriebskräften, der Struktursteifigkeit und der Genauigkeit führen kann (vgl. auch Tabelle 5.1).

Im Verlauf der Analyse konnten die in Abb. 5.26 dargestellten Rekonfigurationsmodule als logische Subsysteme für eine statische Rekonfiguration von Parallelrobotern abgegrenzt werden. In dem Bild sind die auf dem Baukastensystem (vgl. Abb. 5.16) basierenden verschiedenen Systemkomponenten als SysML-Blöcke und die standardisierten Schnittstellen zwischen ihnen als durchgezogene Linien dargestellt. Weiterhin ist ein Informationsfluss zwischen der Steuerung und Systemkomponenten angegeben. Tabelle 5.1 zeigt die Parameter und beeinflussten Anforderungen bezüglich der Rekonfigurationsmodule (vgl. auch [Sch09]). Falls eine Anforderung Ausgangspunkt für eine Rekonfiguration darstellt, können die mit ● markierten Rekonfigurationsmodule angepasst werden. Die Anpassung eines Rekonfigurationsmoduls beeinflusst wie oben beschrieben immer mehrere Anforderungen gleichzeitig. Wo von Beeinflussungen ausgegangen werden kann ist die entsprechende Tabellenzelle gekennzeichnet.

Die Notwendigkeit den Greifer (RM 1) zu ändern, resultiert aus der Beschaffenheit des zu handhabenden Objektes (z.B. Größe, Geometrie, Oberflächenbeschaffenheit) oder dem notwendigen Freiheitsgrad. Dazu wird das Greifprinzip (z.B. Saugnapf, Backengreifer), die Geometrie z.B. der Greifbacken und oder der Greiffreiheitsgrad angepasst. Der neue Greifer muss konstruktiv und steuerungstechnisch in den bestehenden Roboter integriert werden. Besondere Bedeutung

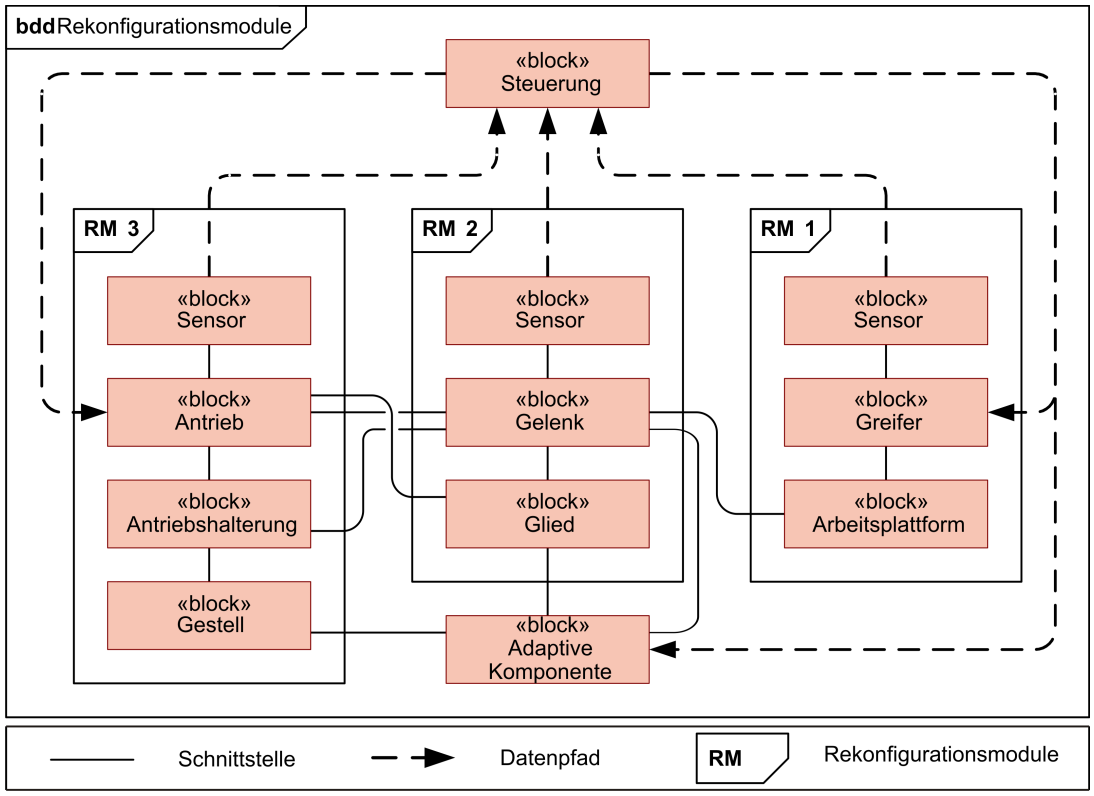


Abbildung 5.26: *Darstellung der Rekonfigurationsmodule (RM 1..3) für die statische Rekonfiguration von Parallelrobotern in einem block definition diagram.*

kommt der Energieversorgung zu, z.B. wenn zusätzliche pneumatische Zuleitungen benötigt werden oder ein elektrischer Antrieb aufgesetzt wird. Durch diese Änderungen wird, durch veränderte Massen und Trägheiten, die Leistungsfähigkeit und die Genauigkeit des Roboters beeinflusst.

Das zweite Rekonfigurationsmodul eignet sich um den Arbeitsraum, Freiheitsgrad, Genauigkeit und Leistungsfähigkeit anzupassen. Die entscheidenden Parameter hierfür sind die kinematische Länge der Glieder und der Winkelbereich den die Gelenke überstreichen können. Eine Änderung dieser Parameter beeinflusst immer auch weitere Anforderungen, da sich die Massen, Trägheiten und Eigenfrequenzen ändern.

Das dritte Rekonfigurationsmodul umfasst im Wesentlichen die Antriebe und deren Platzierung am Gestell. Als Parameter sind hier der Radius auf dem die Antriebe angeordnet werden (vgl. Abb. 5.25), die Drehung der Antriebsebene im Vergleich zur Bezugsebene und die Orientierung der Antriebsachsen zu nennen. Hierbei ist stets auf die Zugänglichkeit des Arbeitsraums zu achten. Der Großteil der betrachteten Anforderungen lässt sich durch eine Anpassung dieser Parameter beeinflussen.

Im Folgenden soll beispielhaft die weitere Rekonfiguration einer einfachen

Tabelle 5.1: *Bei einer Rekonfiguration zu beeinflussende Parameter je Rekonfigurationsmodul (RM) und die Auswirkungen auf Gestaltung und Anforderungen (● Ausgangspunkt, ◐ Beeinflussung, ○ keine Beeinflussung), vgl. [Sch09].*

RM	Parameter	Gestaltungsaspekte	Objekthandhabung	Arbeitsraum	Betriebskräfte	Freiheitsgrad	Genauigkeit	Leistungsfähigkeit	Lebenslaufkosten
1	Greifprinzip, Geometrie, Freiheitsgrad	konstruktive und steuerungstechnische Integration, Energieversorgung	●	○	◐	●	◐	◐	◐
2	Gliedlänge, Gliedprofil, Gelenkwinkelbereich	unterschiedliche kinematische Ketten, Integration von Adaptronik, verschiedene Materialien	○	●	◐	●	●	●	◐
3	Gestellanbindungsradius, Gestellanbindungsebenenwinkel, Antriebsachsenorientierung	Zugänglichkeit	○	●	●	●	●	●	◐

Struktur betrachtet werden. Die Basis der Parameteridentifikation bildet das vorgestellte Modell und das beschriebene Vorgehen. Die konkreten kinematischen Analysen sind nicht mehr Teil der Modellierung. Sie wurden im Teilprojekt A1 des SFB 562 von Mitarbeitern des Institut für Werkzeugmaschinen und Fertigungstechnik der TU Braunschweig durchgeführt [Sch09]. Abb. 5.27 zeigt das kinematische Modell einer **R**$\underline{\mathbf{P}}$**RR**$\underline{\mathbf{P}}$**R** Struktur. Sie besitzt zwei translatorische Freiheitsgrade in der Ebene und wird durch zwei gestellnah angebrachte translatorische Antriebe l_1, l_2 bewegt. Die Struktur ist durch rotatorische Gelenke an den Punkten A und B mit dem Gestell verbunden. Der Abstand zwischen den Anbindungspunkten—der Gestelldurchmesser— wird mit b bezeichnet. Die Neigung der Ebene der Anbindungspunkte wird durch α ausgedrückt, so dass der modifizierte Anbindungspunkt B' entsteht (gestrichelt dargestellte Struktur). Die Position des TCP wird durch die Koordinaten (x_p, y_p) beschrieben.

Durch die oben beschriebenen Parameter können die Lage und die Größe des Arbeitsraums, sowie verschiedene weitere Eigenschaften (z.B. Sensitivität, Steifigkeit) im Arbeitsraum verändert werden. Abb. 5.28 a) zeigt den Einfluss geänderter Gliedlängen l_1, l_2. Der helle Bereich zeigt den Arbeitsraum für kurze Glieder, die in einem Bereich $l = 50..75\,\text{mm}$ angetrieben werden. Der dunkle Bereich zeigt den Arbeitsraum für lange Glieder, die in einem Bereich $l = 75..100\,\text{mm}$ ange-

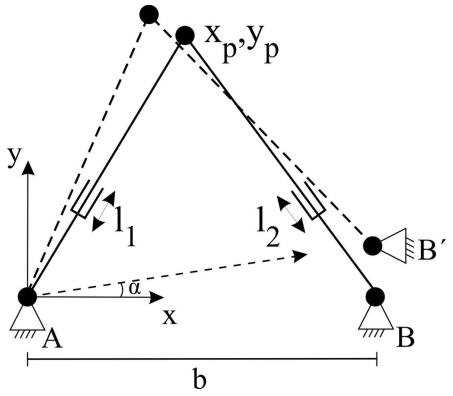


Abbildung 5.27: *Ebene kinematische Struktur mit zwei translatorischen Freiheitsgraden und zwei translatorischen Antrieben, [Sch09].*

trieben werden. Es ist deutlich zu sehen, dass der Arbeitsraum für kurze Glieder deutlich näher am Gestell liegt und in y-Richtung größer, in x-Richtung dagegen kleiner ist. Im Fall $l_1 = l_2 = 50\,\text{mm}$ liegt der TCP auf der x-Achse in einer Singularität vom Typ 2, d.h. die Struktur ist durch die Antriebe nicht eindeutig zu kontrollieren. Es gilt:

$$\det\left(\frac{\partial f}{\partial x}\right) = 0 \ \wedge \ \det\left(\frac{\partial f}{\partial l}\right) \neq 0 \tag{5.5}$$

In Abb. 5.28 b) ist die Modifikation des Gestelldurchmessers b dargestellt. Der helle Bereich zeigt den Arbeitsraum bei einem Gestelldurchmesser $b = 100\,\text{mm}$. Der Arbeitsraum ist etwa quadratisch. Der dunkle Bereich zeigt den Arbeitsraum für einen deutlich verringerten Gestelldurchmesser $b = 40\,\text{mm}$. Dieser Arbeitsraum ist zum einen weiter vom Gestell entfernt und zum anderen entlang der x-Achse deutlich gestreckt. Dabei liegen große Teile des Arbeitsraumes nicht zwischen den Anbindungspunkten.

Schließlich zeigt Abb. 5.28 c) den Einfluss einer Änderung des Gestellanbindungsebenenwinkel α. Der helle Bereich zeigt den Arbeitsraum für eine Ausrichtung der Gestellanbindung entlang der x-Achse, d.h. einem Winkel $\alpha = 0^o$. Der dunkle Bereich zeigt den Arbeitsraum für einen Winkel $\alpha = 20^o$. Der Arbeitsraum verändert dadurch—wie zu erwarten war—weder Form noch Größe, sondern wird nur um den angegebenen Winkel bezüglich des Drehpunktes (hier A) gedreht.

Im Folgenden wird ein mögliches Rekonfigurationsszenario für die oben beschriebene Struktur betrachtet. Ein *pick-and-place*-Roboter soll für eine Montageaufgabe rekonfiguriert werden. Das zu handhabende Objekt und der notwendige Freiheitsgrad bleibt für beide Anwendungsfälle gleich. Für den neuen Anwendungsfall „Montage“ ist ein kleinerer Arbeitsraum ausreichend. Die Betriebskräfte und die Genauigkeit müssen allerdings höher sein. Die Leistungsfähigkeit spielt bei der Montage (d.h. Anzahl von Montagezyklen pro Minute), verglichen mit

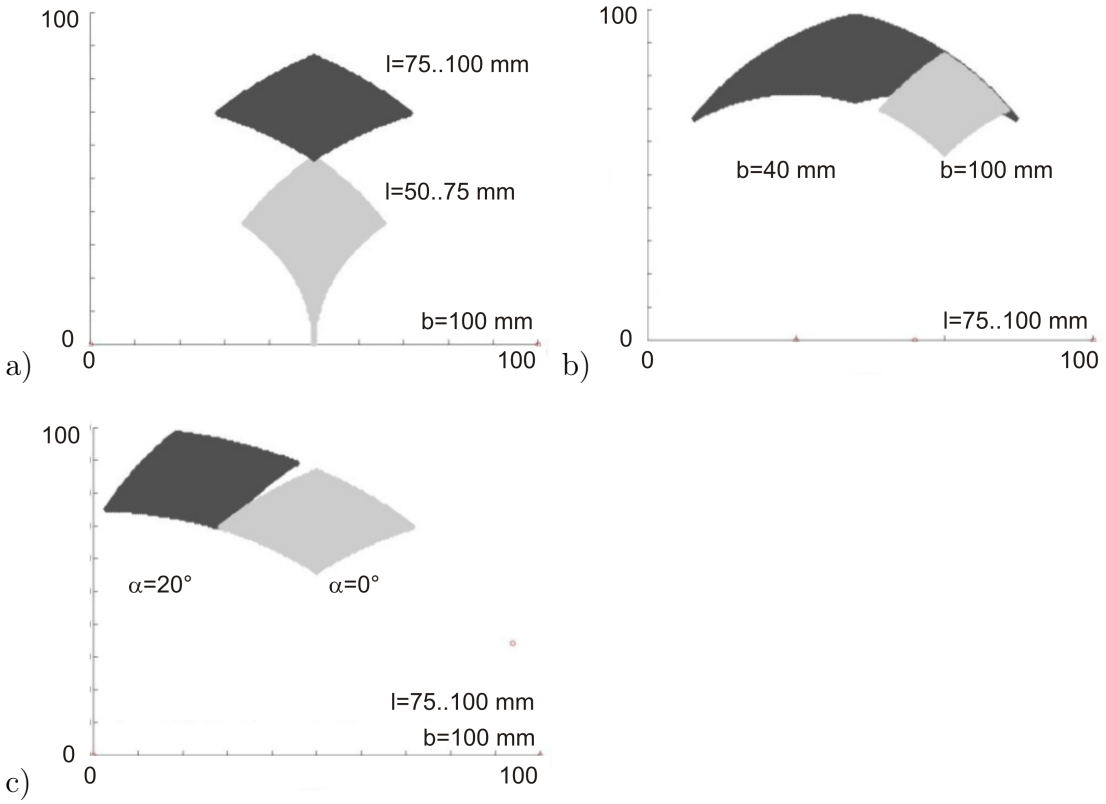


Abbildung 5.28: *Einfluss der Rekonfigurationsparameter auf die Lage und Größe des Arbeitsraums: a) geänderte Gliedlängen l_1, l_2, b) geänderter Gestelldurchmesser b und c) geänderter Gestellanbindungsebenenwinkel α, [Sch09].*

einer *pick-and-place*-Aufgabe eine untergeordnete Rolle. Sie sollte dennoch möglichst hoch sein. Die Betriebskosten sollten möglichst klein bleiben. Aus Tabelle 5.1 wird ersichtlich, dass eine Änderung am Rekonfigurationsmodul 3, spezielle am Gestellradius, einen guten Ausgangspunkt darstellt, um alle betrachteten Anforderungen zu beeinflussen.

In Abb. 5.29 ist ein Ausschnitt aus dem interessierenden Beziehungssystem zu sehen. Es ist gezeigt, dass der neue Anwendungsfall die Anforderungen Betriebskräfte, Genauigkeit und Arbeitsraumdimensionen verfeinert. Die letztgenannte Anforderung konnte direkt dem Parameter *Anbindungsradius* zugeordnet werden (vgl. auch Abb. 5.28 b)). Die Genauigkeit wird über die Sensitivität ebenfalls vom Gestellradius beeinflusst. Die Sensitivität ist hier in x-Richtung für einen größeren Gestellradius prinzipiell besser, d.h. in dieser Richtung kann eine höhere Genauigkeit erzielt werden. Gleichzeitig verschlechtert sich dieser Wert für die y-Richtung. Weiterhin hängt die Genauigkeit von der elastischen Verformung der Struktur ab. Die Größe der Verformung hängt mit der Größe der Betriebskräfte und der strukturellen Steifigkeit zusammen. Bei unveränderter Steifigkeit führen höhere Betriebskräfte zu höheren elastischen Verformungen und damit ei-

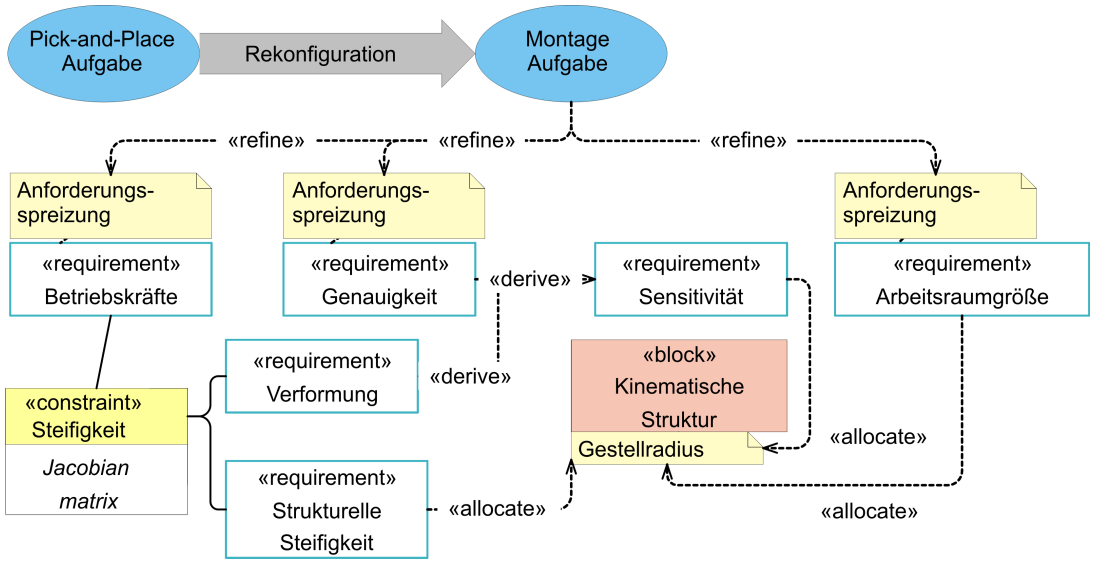


Abbildung 5.29: *Ausschnitt aus dem Beziehungssystem zur Identifikation rekonfigurationsrelevanter Parameter.*

ner schlechteren Genauigkeit. Hier liegt somit ein Zielkonflikt vor, der durch eine Erhöhung der strukturellen Steifigkeit abgemildert werden kann. Die Steifigkeit der Struktur lässt sich ebenfalls durch eine Anpassung des Gestellradius beeinflussen. Ein größerer Gestellradius bewirkt eine höhere Steifigkeit insbesondere in x-Richtung.

Die statische Rekonfiguration wird in diesem Fall also durch eine Vergrößerung des Gestellradius realisiert. Abb. 5.30 a) zeigt den ursprünglichen *pick-and-place*-Anwendungsfall. Auf dem rechten Fließband laufen Drucktaster unsortiert in den Arbeitsraum des Roboters. Auf dem linken Fließband laufen Magazine durch den Arbeitsraum. Der Roboter nimmt jeweils einen Drucktaster auf und legt ihn in einem freien Fach des Magazins ab. Vollständig gefüllte Magazine werden abtransportiert. Hier ist der Gestellradius relativ klein. Der Roboter „lehnt“ sich über sein Gestell hinaus. Durch die hohe Getriebeübersetzung in x-Richtung werden vergleichsweise hohe Geschwindigkeiten erreicht.

Nach der Rekonfiguration des System bietet sich die in Abb. 5.30 b) abgebildete Situation. Auf dem unteren Fließband laufen die Drucktaster in den oben bereits beschriebenen Magazinen in den Arbeitsraum des Roboters. Auf dem oberen Fließband laufen Gehäuse durch den Arbeitsraum. Der Roboter nimmt jeweils einen Drucktaster aus dem Magazin des unteren Fließbandes und setzt ihn in die vorgesehenen Öffnungen des Gehäuses auf dem zweiten Fließband ein. Die Befestigung erfolgt durch an den Drucktastern angebrachte Schnappverbindungen. Bei der Montage sind zum einen Prozesskräfte in y-Richtung notwendig, um die elastische Verformung der Schnapphaken für die Montage zu erreichen. Darüberhinaus sind Kräfte in x-Richtung zu erwarten, da bei der Montage leich-

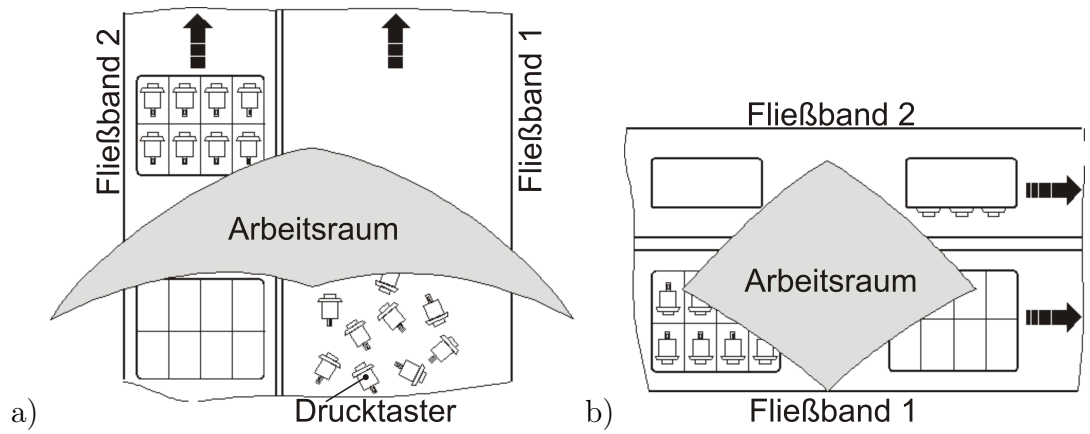


Abbildung 5.30: *Darstellung der Anwendungsfälle a) vor und b) nach Rekonfiguration.*

te Winkelfehler auftreten können. Hier wurde der Gestellradius vergrößert. Der Arbeitsraum wird dadurch zwar insgesamt kleiner, bietet aber eine gleichmäßigere Steifigkeitsverteilung über den Arbeitsraum. Insgesamt erfüllt die neue Konfiguration die Anforderungen aus dem neuen Anwendungsfall deutlich besser. Ein Unternehmen kann den gleichen Roboter, unterschiedlich konfiguriert, an verschiedenen Stationen einsetzen. Damit reduziert sich die Komplexität und das Risiko, denn bei Ausfall eines Roboters kann ggf. eine Station geringer Auslastung stillgelegt werden und die verbleibenden Roboter so rekonfiguriert werden, dass die Produktion nicht eingestellt werden muss. Für eine abschließende Beurteilung müssen jedoch noch detailliertere kinematische und dynamische Untersuchungen durchgeführt werden. Dazu müssen die genauen Produktionsrandbedingungen bekannt sein.

Kapitel 6

Zusammenfassung und Ausblick

Viele Produkte werden heutzutage fachbereichsübergreifend entwickelt. Daraus ergeben sich zahlreiche neue Schwierigkeiten bei der Produktentwicklung. Die Zusammenarbeit von Spezialisten ist auf Grund unterschiedlicher Sichtweisen und „Sprachen“ häufig mit Problemen behaftet. Beispielsweise werden die gleichen Begriffe unterschiedlich gedeutet oder die Fachleute sind nicht in der Lage, ihre Problemstellung so darzustellen, dass sie auch von Nicht-Fachleuten verstanden werden können. Als Folge werden z.B. Schnittstellen nicht ausreichend abgestimmt. Kommen durch die Arbeit in Kooperationsnetzwerken weitere Barrieren hinzu (z.B. räumliche Distanz, Vertragsgrenzen), werden oftmals nur optimierte Teilsysteme entwickelt, die integriert zu suboptimalen Gesamtsystemen führen.

Weitere Schwierigkeiten ergeben sich aus den sich weiter verschärfenden Markterfordernissen, d.h. immer kürzere Markteinführungszeiten, immer häufigere Produktwechsel und ein hoher Kosten und Konkurrenzdruck. Um diese Erfordernisse zu erfüllen, werden Produkte häufig im Wesentlichen konfiguriert und lediglich in Teilbereichen angepasst oder neu entwickelt. Die Erfüllung der Kundenwünsche und der Ziele der Hersteller sowie das Einhalten der Randbedingungen sind die entscheidenden Kriterien für einen nachhaltigen Markterfolg eines Produktes. Bisher gibt es kaum moderne Vorgehensweisen und keine darauf abgestimmten Modellierungskonzepte, um die aktuellen Erfordernisse der Produktentwicklung ausreichend zu berücksichtigen und insbesondere die Modellierung komplexer Anforderungen zu unterstützen.

In dieser Arbeit wurden zunächst die Probleme der interdisziplinären Produktentwicklung systematisch erfasst und aufbereitet. Dazu wurden Produktentwicklungsprozesse der Fachbereiche Maschinenbau, Elektrotechnik und Informationstechnologie aus über fünfzig Jahren Forschung zielgerichtet analysiert. Es wurden wesentliche Gemeinsamkeiten in den Vorgehensweisen identifiziert auch wenn unterschiedliche Begriffe und Sichtweisen das anfangs nicht erwarten ließen. Für die anforderungsbezogene, interdisziplinäre Produktentwicklung konnten wesentliche Bausteine identifiziert werden. Für jeden Baustein wurden die Ansätze aus den verschiedenen Vorgehensweisen zusammengefasst.

Ein weiterer Schwerpunkt der Analyse wurde zudem auf die besonderen Schwierigkeiten der verteilten Produktentwicklung gelegt. Es zeigte sich, dass eine verbesserte Kommunikation zwischen Entwicklungspartnern verschiedener Standorte durch eine bessere rechnerunterstützte Modellierung möglich wird. Modelle die-

nen als Diskussionsgrundlage und Dokumentation. Außerdem zwingen sie zu einem einheitlichen, systematischen Vorgehen.

Schließlich wurden die Besonderheiten im Umgang mit Anforderungen bei der Entwicklung flexibler und modularer Produkte (insbesondere von Baukastensystemen) untersucht. Hier spielt die Identifikation von Zielkonflikten und Anforderungsspreizungen (d.h. Bandbreite der geforderten Werte je Anforderung) eine bedeutende Rolle.

Der letzte Punkt der Analyse befasste sich mit der Bewertung und dem Test der Produkte. Anforderungen bilden hierfür die wesentliche Grundlage. Eine systematische Betrachtung der Anforderungen und Ableitung von Testkriterien und Testfällen hilft die Qualität der Anforderungen zu verbessern und insgesamt ein besseres Produkt zu entwickeln.

Anschließend wurden die für eine Modellierung notwendigen Grundlagen erarbeitet. Dazu wurden zunächst die allgemeinen Anforderungen an Modelle herausgearbeitet und die verschiedenen notwendigen Partialmodelle der Produktentwicklung identifiziert. Die SysML wurde als sinnvoll zu verwendende Beschreibungssprache identifiziert und ihr Aufbau, sowie die Möglichkeiten von kommerziellen Softwarewerkzeugen diskutiert.

Besonderen Stellenwert bei der Produktentwicklung bildet neben einer systematischen Modellierung die zielgerichtete Auswertung der erstellten Modelle. Für die Auswertung ist neben der Darstellung in Diagrammen, Tabellen und Matrizen die Klassifikation und Bildung von beschreibenden Kennzahlen von Bedeutung. Der Schwerpunkt lag in der Aufbereitung von Beziehungen, die für die Auswertung des Modells verwendet werden. Es können verschiedene Inter- und Intramodellbeziehungen unterschieden werden, die unterschiedliche Möglichkeiten zur Modellauswertung bereitstellen.

Auf Basis der erarbeiteten Grundlagen wurde ein Vorgehen zur Produktentwicklung vorgeschlagen, welches durch ein detailliertes Modellierungskonzept unterstützt wird. Zunächst wurden sämtliche notwendigen Partialmodelle mit allen Modellierungsobjekten vorgestellt. Außerdem wurden die Vorgehensweisen und die Strukturierung innerhalb der Partialmodelle erläutert. Des Weiteren wurden die Inter- und Intramodellbeziehungen für jedes Partialmodell genau betrachtet. Im weiteren Verlauf wurden die entwickelten Werkzeuge zur Modellauswertung im Einzelnen vorgestellt und detailliert beschrieben. Es konnte z.B. gezeigt werden, wie Zielkonflikte gefunden werden können und wie das Projekt durch das Erfassen von Kennzahlen überwacht werden kann. Schließlich wurde ein spezielles Vorgehen zur Verwendung der erarbeiteten Modellierung bei der Entwicklung von Baukastensystemen vorgeschlagen.

Die Akzeptanz des erarbeiteten Konzepts wurde in einem Feldversuch anhand studentischer Projekte untersucht. Diese ersten Befragungsergebnisse lassen darauf schließen, dass das Konzept für eine industrielle Anwendung in der Produktentwicklung grundsätzlich geeignet ist und dort zu guten Ergebnissen führen kann. Es zeigte sich aber auch, dass das Konzept für Produktentwicklungen ge-

ringer Komplexität überdimensioniert ist und das Nutzen-Aufwand-Verhältnis in solchen Fällen zu klein ausfällt.

Des Weiteren wurde im Rahmen des SFB 562 ein umfangreiches Modell für das komplexe Produkt „Parallelroboter“ erarbeitet. Dazu wurden sämtliche Partialmodelle erstellt und die zuvor beschriebenen Auswertemechanismen angewendet. Trotz des mit über 400 Anforderungen noch vergleichsweise geringen Umfangs, konnten durch die systematische Auswertung einige neue Beziehungen gefunden werden. Diese stellen Anforderungen derart in Beziehung, dass es möglich wird zu beurteilen, ob zwei Anforderungen einander unterstützen (z.B. „*hohe Beschleunigung des Endeffektors ermöglichen*“ unterstützt „*hohe Endgeschwindigkeit am Endeffektor erreichen*“) oder in einem Konflikt stehen (z.B. „*kleines Gelenkspiel erreichen*“ steht in Konflikt zu „*geringe Gelenkreibung erreichen*“). Auf Basis des Beziehungssystem der Anforderungen lassen sich schließlich Zielkonflikte identifizieren (z.B. „*hohe Dynamik erzielen*“ steht im Konflikt zu „*hohe Genauigkeit erzielen*“).

Abschließend wurde das erarbeitete Modell für die Identifikation von rekonfigurationsrelevanten Parametern von Parallelrobotern eingesetzt. Es konnten die wesentlichen Parameter für eine sinnvolle statische Rekonfiguration von Parallelrobotern gefunden werden. Die Ergebnisse wurden anhand einer einfachen kinematischen Struktur näher untersucht.

Es konnte gezeigt werden, dass das entwickelte Modellierungskonzept die Bearbeiter unterstützt, die gesetzten Ziele zu erreichen. Die Ergebnisse des Feldversuchs deuten auf eine breite Akzeptanz hin und die systematische, ganzheitliche Betrachtung lässt auch in anderen Branchen gute Ergebnisse erwarten (vgl. [Rie10]). Bedingt durch den weiten betrachteten Bereich konnte nicht überall gleichermaßen in die Tiefe gearbeitet werden. Es bleiben offene Fragen, die in weiteren Arbeiten behandelt werden müssen.

Alle in dieser Arbeit gezeigten Makros zur Modellauswertung sind prototypisch umgesetzt worden. Für eine nutzerfreundliche Auswertung des Modells müssen die Makros hinsichtlich Rechenzeit, Benutzerführung und Aussehen optimiert werden. Zudem ist keine bidirektionale Kommunikation mit dem Modellierungswerkzeug möglich. Änderungen, die in Microsoft Excel gemacht werden, können nicht automatisch in das Artisan Studio Modell übernommen werden. Hierzu sind vor allem weitere Arbeiten seitens der Hersteller notwendig.

Eine verbesserte Unterstützung der Auslegung in den frühen Phasen kann durch eine weitere Anbindung von mathematischen Werkzeugen erreicht werden. Wäre es beispielsweise möglich, die im Artisan-Modell erzeugten *Constraints* gezielt an ein CAS zu übergeben, so könnten bereits in frühen Phasen sehr schnell Vergleichs- und Optimierungsrechnungen durchgeführt werden. Eine Verknüpfung der bereits beschriebenen Werkzeuge mit CAD-Systemen kann zudem zu einer weiteren Beschleunigung des Entwicklungsprozesses führen. Beispielsweise können die im Artisan-Modell identifizierten Rekonfigurationsparameter im parametrisch aufgebauten CAD-Modell zur Steuerung dienen. Gleichzeitig kön-

nen Auslegungsrechnungen im CAS-Modell erfolgen, die über bereits vorhandene Schnittstellen zum CAD (z.B. Maple und SolidWorks) halbautomatisch optimierte Geometrien erzeugen.

Weiterhin sollte das Konzepts um eine Betrachtung von Kostengesichtspunkten erweitert werden. Für eine vollständige Beurteilung von Produkten müssen neben den technischen auch wirtschaftliche Aspekte detailliert betrachtet werden. Erste Arbeiten lassen darauf schließen, dass die Integration eines Kostenpartialmodells möglich und sinnvoll ist (vgl. [Mei09]). Dazu können die Anwendungsfälle des Produktlebenslaufs im Sinne einer Prozesskostenrechnung und/oder die im Systemkontext modellierten Komponenten im Sinne einer vereinfachten Kostenermittlung (vgl. [VDI98]) herangezogen werden. Wichtig ist, dass hier ebenfalls eine ganzheitliche lebenslauforientierte Betrachtung erfolgt. Insbesondere bei Investitionsgütern nehmen die Betriebskosten oft den weitaus größten Anteil an den gesamten Lebenslaufkosten ein.

Schließlich sollten im Sinne der empirischen Konstruktionsforschung Analysen folgen, die weit über die generelle Akzeptanzuntersuchung dieser Arbeit hinausgehen. Dadurch können die verschiedenen Einflüsse auf die Entwicklung besser eingeschätzt und das Denken und Verhalten der Probanden bei der Entwicklung besser verstanden werden. Dies ermöglicht eine gezielte Verbesserung der Anforderungsmodellierung zur Unterstützung der Mitarbeiter in interdisziplinären Produktentwicklungsprojekten.

Literaturverzeichnis

[Ada92] Adami, W. *Strukturen wissensbasierter Systeme für die rechnergestützte Konstruktion.* Vulkan Verlag, Dissertation, Institut für Werkzeugmaschinen und Fertigungstechnik der Technischen Universität Braunschweig, Essen, ISBN 3-8027-8609-2, 1992.

[Ahr00] Ahrens, G. *Das Erfassen und Handhaben von Produktanforderungen - Methodische Voraussetzungen und Anwendung in der Praxis.* Deutsche Nationalbibliothek, Dissertation, Bereich Konstruktionstechnik am Institut für Maschinenkonstruktion der Technischen Universität Berlin, Frankfurt am Main, 2000.

[Alt84] Altschuller, G.S. *Erfinden: Wege zur Lösung technischer Probleme.* VEB Verlag Technik, Berlin, 1984.

[Alv09] Alvarez Cabrera, A.A.; Erden, M.S.; Tomiyama, T. On the Potential of Function-Behavior-State (FBS) Methodology for the Integration of Modeling Tools. In *Cirp Design Conference - Design Synthesis.* Cranfield/ UK, ISBN 978-0-9557436-4-1, S. 412-419, 30.-31. März 2009.

[Ang08] Anggreeni, I.; van der Voort, M.C. Classifying Scenarios in a Product Design Process: a study towards semi-automated scenario generation. In *Cirp Design Conference - Design Synthesis.* Enschede/ Niederlande, ISBN 978-90-365-2634-0, S. 54, 7.-9. April 2008.

[Ari07] Ariyo, O.O.; Keller, R.; Eckert, C.M.; Clarkson, P.J. Predicting Change Propagation on Different Levels of Granularity: An Algorithmic View. In *16th International Conference on Engineering Design - ICED 07.* Paris/ Frankreich, ISBN 1-904670-02-4, ID 220, 28.-31. August 2007.

[Art10] Artisan Software Tools. *Homepage der Firma Artisan Software Tools.* http://www.artisansoftwaretools.com/, *zuletzt Aufgerufen am 26.1.10*, 2010.

[Avg07] Avgoustinov, N. *Modelling in Mechanical Engineering and Mechatronics : Towards Autonomous Intelligent Software Models.* Springer Verlag, London, ISBN 978-1-8462-8908-8, 2007.

[Ban94] Bandler, R. *Metasprache und Psychotherapie: Die Struktur der Magie I.* Jungfermann, Paderborn, ISBN 3-87387-186-6, 1994.

[Bec00] Beck, K. *eXtreme Programming Explained.* Addison-Wesley, München, ISBN 3-8273-1709-6, 2000.

[Bec06] Becker, A. Erstellung eines Baukastenkonzepts für die Gestelle von Parallelrobotern. unveröffentlichte Studienarbeit, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 2006.

[Ben01] Bender, B. *Zielorientiertes Kooperationsmanagement in der Produktentwicklung.* Verlag Dr. Hut, Dissertation, Fachgebiet Produktentwicklungsmethodik der Technischen Universität München, München, ISBN 3-934767-31-1, 2001.

[Ber00] Berhard, R. Entwicklungstrends in der Robotik. *ZWF - Zeitschrift für den wirtschaftlichen Fabrikbetrieb*, 95(3):88–93, 2000.

[Bid68] Bidlingmaier, J. *Zielkonflikte und Zielkompromisse im unternehmerischen Entscheidungsprozeß.* Betriebswirtschaftlicher Verlag Dr. Th. Gabler, Wiesbaden, 1968.

[Bie06] Bier, C.C. *Geometrische und physikalische Analyse von Singularitäten bei Parallelstrukturen.* Vulkan Verlag, Dissertation, Institut für Werkzeugmaschinen und Fertigungstechnik der Technischen Universität Braunschweig, Essen, ISBN 978-3-8027-8690-7, 2006.

[Bir80] Birkhofer, H. *Analyse und Synthese der Funktionen technischer Produkte.* VDI-Verlag GmbH, Dissertation, Institut für Konstruktionslehre, Maschinen- und Feinwerkelemente der Technischen Universität Braunschweig, Düsseldorf, ISBN 3-18-147001-5, 1980.

[Bir91] Birkhofer, H. Methodik in der Konstruktionspraxis - Erfolge, Grenzen und Perspektiven. In *Proc. of International Conference on Engineering Design ICED'91.* Heurista Verlag, Zürich/Schweiz, ISBN 3-85693-024-8 S. 224-233, 27.-29. August 1991.

[Bis07] Bischof, A.; Blessing, L. Gestaltungsrichtlinien zur Entwicklung flexibler Produkte. In *18. Symposium Design for X.* Neukirchen, ISBN 978-3-9808539-5-8, S. 1-12, 11.-12. Oktober 2007.

[Ble08a] Blees, C.; Jonas, H.; Krause, D. Entwurf von modularen Produktarchitekturen unter Betrachtung unterschiedlicher Unternehmenssichten. In *Design for X.* Neukirchen, ISBN 978-3-9808539-6-5, S. 149–158, 9.-10. Oktober 2008.

[Ble08b] Blees, C.; Krause, D. On the Development of Modular Product Structures: A Differentiated Approach. In *International Design Conference*. Dubrovnik/ Kroatien, ISBN 978-953-6313-93-8, S. 301-308, 19.-22. Mai 2008.

[Boe88] Boehm, B.W. A Spiral Model of Software Development and Enhancement. *Computer*, 21(5):61–72, 1988.

[Bor61] Borowski, K.-H. *Das Baukastensystem in der Technik.* Springer Verlag, Berlin, 1961.

[Bre93] Breiing, A.; Flemming, M. *Theorie und Methoden des Konstruierens.* Springer-Verlag, 1993.

[Bre03] Brey, M. *Konfiguration und Gestaltung mit Constraintsystemen.* Cuvillier Verlag, Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, Göttingen, ISBN 3-89873-643-1, 2003.

[Büt05] Büttgenbach, S.; Güttler, J.; Last, P.; Bier, C.; Otremba, R. Development of Angular Joint-Sensors and Application to Parallel Robots. In *Proc. of 2nd International Colloquium of the Collaborative Research Center 562.* Braunschweig, S. 237-251, 2005.

[Büt07] Büttgenbach, S. (Hrsg.). *Sonderforschungsbereich 516–Antrag auf Finanzierung und Ergebnisbericht.* SFB 516, Braunschweig, 2007.

[Bud09] Budde, C. *Wechsel der Konfiguration zur Arbeitsraumvergrößerung bei Parallelrobotern.* Vulkan Verlag, Dissertation, Institut für Werkzeugmaschinen und Fertigungstechnik der Technischen Universität Braunschweig, Essen, ISBN 978-3-8027-8747-8, 2009.

[Bun04] Bundesrepublik Deutschland. *Dokumentation des V-Modell XT, Version 1.1.0.* http://h90761.serverkompetenz.net/v-modell-xt/Release-1.1/Dokumentation/html/, *zuletzt Aufgerufen am 18.8.08*, 2004.

[Cha08] Chahadi, Y.; Birkhofer, H. Begriffssystem zur Unterstützung der automatisierten Aufgabenklärung (Anforderungsermittlung). In *19. Symposium Design for X.* Neukirchen, ISBN 978-3-9808539-6-5, S. 129-138, 9.-10. Oktober 2008.

[Czi06] Czichos; H. *Mechatronik - Grundlagen und Anwendungen technischer Systeme.* Vieweg Verlag, Wiesbaden, ISBN 3-8348-0171-2, 2006.

[Dae02] Daenzer, W.F.; Huber, F. (Hrsg.). *Systems Engineering - Methodik und Praxis.* Verlag industrielle Organisation, 2002.

[Dan96] Danner, S. *Ganzheitliches Anforderungsmanagement für marktorientierte Entwicklungsprozesse.* Shaker Verlag, Dissertation, Lehrstuhl für Konstruktion im Maschinenbau der Technischen Universität München, Aachen, ISBN 3-8265-1908-6, 1996.

[Dei07] Deimel, M. *Ähnlichkeitskennzahlen zur systematischen Synthese, Beurteilung und Optimierung von Konstruktionslösungen.* VDI Verlag, Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, Düsseldorf, ISBN 978-3-18-339801-0, 2007.

[Des73] Descartes, R. *Regeln zur Ausrichtung der Erkenntniskraft.* Felix Meiner Verlag, Hamburg, ISBN 3-7873-0265-4, 1973.

[Deu08] Deutsche Forschungsgemeinschaft. *Offizielle Homepage der DFG.* http://www.dfg.de/, *zuletzt Aufgerufen am 16.10.08*, 2008.

[DIN00] DIN 743. *Tragfähigkeitsberechnung von Wellen und Achsen.* Beuth Verlag, Berlin, 2000.

[DIN02] DIN EN 12973. *Value Management.* Beuth Verlag, Berlin, 2002.

[DIN05] DIN EN ISO 9000. *Qualitätsmanagementsysteme - Grundlagen und Begriffe.* Beuth Verlag, Berlin, 2005.

[Dre93] Drebing, U. *Zur Metrik der Merkmalsbeschreibung für Produktdarstellende Modelle beim Konstruieren.* Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 1993.

[Ehr05] Ehrlenspiel, K.; Kiewert, A.; Lindemann, U. *Kostengünstig Entwickeln und Konstruieren : Kostenmanagement bei der integrierten Produktentwicklung.* Springer Verlag, Berlin, ISBN 978-3-540-25165-1, 2005.

[Ehr07] Ehrlenspiel, K. *Integrierte Produktentwicklung - Denkabläufe, Methodeneinsatz, Zusammenarbeit.* Carl Hanser Verlag, 2007.

[Eil99] Eiletz, R. *Zielkonfliktmanagement bei der Entwicklung komplexer Produkte - am Bsp. PKW-Entwicklung.* Shaker Verlag, Dissertation, Lehrstuhl für Konstruktion im Maschinenbau der Technischen Universität München, Aachen, ISBN 3-8265-6019-1, 1999.

[Fin05] Finkemeyer, B.; Kröger, T.; Wahl, F. Aktionsprimitive: Ein universelles Roboter-Programmierparadigma. *at - Automatisierungstechnik*, (04):189–196, 2005.

[Fir03] Firchau, N.L. *Variantenoptimierende Produktgestaltung.* Cuvillier Verlag, Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, Göttingen, ISBN 3-89873-934-1, 2003.

[Fra75] Franke, H.-J. Methodische Schritte beim Klären konstruktiver Aufgabenstellungen. *Konstruktion*, (27):395–402, 1975.

[Fra76] Franke, H.-J. *Untersuchungen zur Algorithmisierbarkeit des Konstruktionsprozesses*. VDI Verlag, Dissertation, Institut für Konstruktionslehre, Maschinen- und Feinwerkelemente der Technischen Universität Braunschweig, Düsseldorf, ISBN 3-18-144701-3, 1976.

[Fra85] Franke, H.-J. Konstruktionsmethodik und Konstruktionspraxis - eine kritische Betrachtung. In *International Conference on Engineering Design - ICED 85*. Hamburg, ISBN 3-85693-014-0, S. 910-924, 26.-29. August 1985.

[Fra98] Franke, H.-J. Design Methods before the Change of Paradigms. In *Universal Design Theory, Grabowski, H. (Hrsg.)*. Shaker Verlag, Aachen, ISBN 3-8265-4265-7, S. 249-269, 1998.

[Fra01] Franke, H.-J.; Brey, M.; Jänicke, T. „eKat“ - Rechnerunterstütztes Konstruktionskatalogsystem. In *Manufacturing '01*. Poznan/ Polen, S. 63-70, 8.-9. November 2001.

[Fra02a] Franke, H.-J.; Brey, M.; Jänicke, T.; Wrege, C. Verarbeitung von Anforderungen in der standortübergreifenden Produktentwicklung. *ZWF - Zeitschrift für den wirtschaftlichen Fabrikbetrieb*, 97(12):606–609, 2002.

[Fra02b] Franke, H.-J.; Hesselbach, J.; Huch, B.; Firchau, N.L. *Variantenmanagement in der Einzel- und Kleinserienfertigung*. Carl Hanser Verlag, München, ISBN 978-3-446-21730-4, 2002.

[Fra02c] Franke, H.-J.; Otremba, R.; Jänicke, T. Methodical Development of Optimized Passive Joints. In *Proc. of the 1st International Colloquium of the Collaborative Research Centre 562 „Robotic Systems for Handling and Assembly“*. Krefft, M.; Wahl, F. (Hrsg.), Shaker, Aachen, Braunschweig, ISBN 978-3-8322-0201-9, S. 119-130, 29.-30. Mai 2002.

[Fra04] Franke, H.J.; Löffler, S.; Deimel, M. Increasing the Efficiency of Design Catalogues by Using Modern Data Processing Technologies. In *Proc. of International Design Conference*. Dubrovnik/ Kroatien, ISBN 953-6313-60-X, S. 853–858, 18.-21. Mai 2004.

[Fra05a] Franke, H.-J.; Deimel, M. Zusammenhänge zwischen Funktion und topologischer Struktur. In *Proc. of 50. Internationales Wissenschatliches Kolloquium*. Ilmenau, ISBN 3-932633-98-9, S. 1-20, 19.-23. September 2005.

[Fra05b] Franke, H.-J.; Huch, B.; Herrmann, C.; Löffler, S. (Hrsg.). *Kooperationsorientiertes Innovationsmanagement - Ergebnisse des BMBF-Verbundprojektes GINA „Ganzheitliche Innovationsprozesse in modularen Unternehmensnetzwerken".* Logos Verlag, Berlin, ISBN 978-3-8325-0828-9, 2005.

[Fra05c] Franke, H.-J.; Wrege, C.; Stechert, C. Werkzeuge für die Parallelroboterentwicklung - Entwicklungsumgebung und Wissensmanagement im SFB 562 an der TU Braunschweig. *wt Werkstatttechnik online*, (9):694–698, 2005.

[Fra05d] Franke, H.-J.; Wrege, C.; Stechert, C.; Pavlovic, N. Knowledge Based Development Environment. In *The 2nd International Colloquium of the Collaborative Research Center 562.* Shaker Verlag, Braunschweig, ISBN 3-8322-3866-2, S. 221-236, 10.-11. Mai 2005.

[Fra06] Frank, U. *Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme.* Heinz-Nixdorf-Institut, Dissertation, Fachgebiet Rechnerintegrierte Produktion am Heinz Nixdorf Institut der Universität Paderborn, Paderborn, ISBN 3-935433-84-0, 2006.

[Fra08] Franke, H.-J.; Pavlovic, N.; Kirchhoff, M.R.; Büttgenbach, S. Integration of Sensors and Actuators in Joints of Parallel Robots. In *Proc. of 3rd International Colloquium of the Collaborative Research Center 562.* Braunschweig, ISBN 978-3-8322-7129-9, S. 169-180, 28.-29. April 2008.

[Fra09] Franke, H.-J. *Grundlagen der Produktentwicklung und Konstruktion.* Umdruck zur Vorlesung am Institut für Konstruktionstechnik der TU Braunschweig, 2009.

[Fri90] Frick, R. Arbeit des Industrial Designers im Entwicklungsteam. *Konstruktion*, (42):149–156, 1990.

[Fri01] Frindt, M. *Modulbasierte Synthese von Parallelstrukturen für Maschinen in der Produktionstechnik.* Vulkan Verlag, Dissertation, Institut für Werkzeugmaschinen und Fertigungstechnik der Technischen Universität Braunschweig, Essen, ISBN 3-8027-8659-9, 2001.

[Gah04] Gahr, A.; Lindemann, U. Ein methodischer Ansatz zur induzierten Kostenkalkulation individualisierter Produkte. In *15. Symposium Design for X.* Neukirchen, ISBN 3-9808539-2-6, S. 159-168, 14.-15. Oktober 2004.

[Gaj88] Gajski, D.D. *Silicon Compilation.* Addison-Wesley Publishing Company, 1988.

[Gau01] Gaul, H.-D. *Verteilte Produktentwicklung: Perspektiven und Modell zur Optimierung.* Verlag Dr. Hut, Dissertation, Lehrstuhl für Produktentwicklung der Technischen Universität München, München, ISBN 3-934767-36-2, 2001.

[Gau08a] Gausemeier, J.; Frank, U.; Donoth, J.; Kahl, S. Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme des Maschinenbaus (Teil 1). *Konstruktion*, 60(7/8):59–66, 2008.

[Gau08b] Gausemeier, J.; Frank, U.; Donoth, J.; Kahl, S. Spezifikationstechnik zur Beschreibung der Prinziplösung selbstoptimierender Systeme des Maschinenbaus (Teil 2). *Konstruktion*, 60(9):91–99, 2008.

[Goh98] Gohritz, A. Anforderungsbild an parallelstrukturgerechte Baugruppen. In *Chemnitzer Parallelstruktur-Seminar.* Verlag wissenschaftliche Scripten, Zwickau, ISBN 3-928921-35-5, S. 15-26, 1998.

[Gör01] Göring, O. Portalroboter aus dem Baukasten. *Maschinenmarkt*, 107(48):54–55, 2001.

[Gra93] Grabowski, H.; Anderl, R.; Polly, A. *Integriertes Produktmodell.* Beuth Verlag, Berlin, ISBN 3-410-12920-0, 1993.

[Gun08] Gunzenhauser, M. *Platform Concepts for the Systems Business: Design and Development of Global Product Platforms.* VDI Verlag, Dissertation, Institut für Werkzeugmaschinen und Fertigung der ETH Zürich, Düsseldorf, ISBN 978-3-18-319116-1, 2008.

[GWJ10] GWJ Technology. *Homepage des Herstellers des eAssistant.* http://www.eassistant.eu/, *zuletzt Aufgerufen am 30.1.10*, 2010.

[Hes00] Hesselbach, J.; Kusiek, A. Maschinennahe Steuerungsfunktionen für Parallelroboter. In *Proc. of 2nd Chemnitz Parallel Kinematics Seminar.* Chemnitz, S. 221-240, 2000.

[Hob88] Hoberg, R. et al. *Der kleine Duden - Deutsche Grammatik.* Dudenverlag, Mannheim, ISBN 3-411-02182-9, 1988.

[Hub92] Hubka, V.; Eder, W.E. *Einführung in die Konstruktionswissenschaft - Übersicht, Modell, Ableitungen.* Springer-Verlag, 1992.

[Hum95] Humpert, A. *Methodische Anforderungsverarbeitung auf Basis eines objektorientierten Anforderungsmodells.* HNI-Verlagsschriftenreihe, Dissertation, fachgebiet Rechnerintegrierte Produktion am Heinz Nixdorf Institut der Universität Gesamthochschule Paderborn, Paderborn, ISBN 3-931466-08-6, 1995.

[IEE98a] IEEE Standard 1233. *IEEE Guide for Developing System Requirements Specifications.* IEEE Software Engineering Standards Committee of the IEEE Computer Society, New York, 1998.

[IEE98b] IEEE Standard 830. *IEEE Recommended Practice for Software Requirements Specifications.* IEEE Software Engineering Standards Committee of the IEEE Computer Society, New York, 1998.

[IEE08] IEEE Standard 829. *IEEE Standard for Software and System Test Documentation.* IEEE Software and Systems Engineering Standards Committee of the IEEE Computer Society, New York, 2008.

[Jes96] Jeschke, A. *Beitrag zur wirtschaftlichen Bewertung von Standardisierungsmaßnahmen in der Einzel- und Kleinserienfertigung durch die Konstruktion.* Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 1996.

[Jän07] Jänsch, J. *Akzeptanz und Anwendung von Konstruktionsmethoden im industriellen Einsatz - Analyse und Empfehlungen aus kognitionswissenschaftlicher Sicht.* VDI-Verlag GmbH, Dissertation, Fachgebiet Produktentwicklung und Maschinenelemente der Technischen Universität Darmstadt, Düsseldorf, ISBN 978-3-18-339601-6, 2007.

[Joh08] Johar, A.; Stetter, R. A Proposal for the Use of Diagrams of UML for Mechatronics Engineering. In *10th International Design Conference.* Dubrovnik/ Kroatien, ISBN 978-953-6313-93-8, S. 1287-1294, 19.-22. Mai 2008.

[Jov02] Jovane, F. et al. Design issues for reconfigurable PKMs. In *4th Chemnitz Parallel Kinematic Seminar.* Verlag wissenschaftliche Scripten, Zwickau, ISBN 978-3-928921-76-3, S. 69-82, 2002.

[Jun88] Junge, H. Roboter aus dem Baukasten. *TECHNICA*, (23):17–20, 1988.

[Jun06] Jung, C. *Anforderungsklärung in interdisziplinärer Entwicklungsumgebung.* Verlag Dr. Hut, Dissertation, Lehrstuhl für Produktentwicklung der Technischen Universität München, München, ISBN 978-3-89963-367-2, 2006.

[Kan84] Kano, N. Attractive Quality and Must Be Quality. *Quality*, 14(2):39–48, 1984.

[Kei08] Keimer, R.; Algermissen, S.; Pavlovic, N.; Rose, M. Active Vibration Suppression in Parallel Mechanisms for Handling and Assembly. In *Proc. of 19th International Conference on Adaptive Structures and Technologies (ICAST).* Ascona/Schweiz, 6.-9. Oktober 2008.

[Ker98] Kerle, H.; Frindt, M.; Plitea, N. Zur Systematik von Maschinen und Geräten mit Parallelstruktur. In *Proc. of VDI-Fachtagung - Kurvengetriebe, Koppelgetriebe und gesteuerte Antriebe.* Kassel, S. 209-224, 1998.

[Kes54] Kesselring, F. *Technische Kompositionslehre.* Springer-Verlag, 1954.

[Küh68] Kühnpast, R. *Das System der selbsthelfenden Lösungen in der maschinenbaulichen Konstruktion.* Dissertation, Lehrstuhl und Laboratorium I für Maschinenelemente der Technischen Hochschule Darmstadt, 1968.

[Kic95] Kickermann, H. *Rechnerunterstützte Verarbeitung von Anforderungen im methodischen Konstruktionsprozeß.* Dissertation, Institut für Konstruktionslehre, Maschinen- und Feinwerkelemente der Technischen Universität Braunschweig, Braunschweig, 1995.

[Kir09] Kirchner, K.; Drebing, U.; Franke, H.-J. Konstruktionskataloge für den effizienten Einsatz physischer Modelle im Produktentwicklungsprozess. *Konstruktion*, (10):62–66, 2009.

[Klä93] Kläger, R. *Modellierung von Produktanforderungen als Basis für Problemlösungsprozesse in intelligenten Konstruktionssystemen.* Shaker Verlag, Dissertation, Institut für Rechneranwendung in Planung und Konstruktion der Universität Karlsruhe, Aachen, ISBN 978-3-86111-622-6, 1993.

[Kle04] Klein, B. Abbau von Überkomplexität in Produkten und Prozessen. *Konstruktion*, 56(10):75–81, 2004.

[Kno07] Knollmann, V. *UML-basierte Testfall- und Systemmodelle für Eisenbahnleit- und - sicherungstechnik.* Dissertation, DLR-Institut für Verkehrssystemtechnik an der Technischen Universität Braunschweig, ISBN 978-3-00-024113-0, 2007.

[Koh97] Kohlhase, N. *Strukturieren und Beurteilen von Baukastensystemen: Strategien, Methoden, Instrumente.* VDI Verlag, Dissertation, Technische Hochschule Darmstadt, Düsseldorf, ISBN 3-18-327501-5, 1997.

[Kol94] Koller, R. *Konstruktionslehre für den Maschinenbau.* Springer-Verlag, 1994.

[Kon98] Konhäuser, W. *Industrielle Steuerungstechnik: Grundlagen und Anwendungen.* Carl Hanser Verlag, München, ISBN 3-446-19368-5, 1998.

[Koo06] Koordinierungs- und Beratungsstelle der Bundesregierung für Informationstechnik in der Bundesverwaltung. *UfAB IV - Unterlage für*

Ausschreibung und Bewertung von IT-Leistungen - Version 1.0. Bundesministerium des Inneren, Berlin, 2006.

[Kor99] Koren, Y.; Heisel, U.; Jovane, F.; Moriwaki, T.; Pritschow, G.; Ulsoy, G.; Van Brussel, H. Reconfigurable Manufacturing Systems. *Annals of the CIRP*, 48(2):527–540, 1999.

[Küp93] Küpper, H.-U. Beschaffung. In *Vahlens Kompendium der Betriebswirtschaftslehre*. Bitz, M.; Dellmann, K.; Domsch, M.; Egner, H.(Hrsg.), Vahlen, München, S. 203-262, 1993.

[Krä07] Krämer, S. *Total Cost of Ownership: Konzept, Anwendung und Bedeutung im Beschaffungsmanagement deutscher Industrieunternehmen.* VDI Verlag, Saarbrücken, ISBN 978-3-8364-1933-8, 2007.

[Kre06a] Krefft, M. *Aufgabenangepasste Optimierung von Parallelstrukturen für Maschinen in der Produktionstechnik*. Vulkan Verlag, Dissertation, Institut für Werkzeugmaschinen und Fertigungstechnik der Technischen Universität Braunschweig, Essen, ISBN 3-8027-8689-0, 2006.

[Kre06b] Krefft, M.; Brüggemann, H.; Herrmann, C.; Hesselbach, J. Reconfigurable Parallel Robots: Combining High Flexibility and Short Cycle Times. *Journal of Production Engineering*, 13(1):109–112, 2006.

[Kru96] Kruse, P. J. *Anforderungen in der Systementwicklung - Erfassung, Aufbereitung und Bereitstellung von Anforderungen in interdisziplinären Entwicklungsprojekten*. VDI-Verlag, Dissertation, Institut für Maschinenwesen der Technischen Universität Clausthal, Düsseldorf, ISBN 3-18-319120-2, 1996.

[Kru00] Krusche, T. *Strukturierung von Anforderungen für eine effiziente und effektive Produktentwicklung*. Verlag Mainz, Wissenschaftsverlag, Dissertation, Institut für Konstruktionslehre, Maschinen- und Feinwerkelemente der Technischen Universität Braunschweig, Aachen, ISBN 3-89653-796-2, 2000.

[La 06] La Rocca, G.; van Tooren, M.J.L. A modular reconfigurable software tool to support distributed multidisciplinary design and optimisation of complex products. In *16th CIRP International Design Seminar*. Kananaskis/ Kanada, 16.-19. Juli 2006.

[Las08] Last, P.; Raatz, A.; Hesselbach, J.; Pavlovic, N.; Keimer, R. Parallel Robot Calibration Utilizing Adaptronic Joints. In *Proc. of ASME 32nd Annual Mechanisms and Robotics Conference*. New York/USA, 3.-6. August 2008.

[Löh91] Löhnert, H.; Müller, M. *Produktprofil-Planung - Vorgehensweisen, Methoden, Tool.* Schriftenreihe Blaue Broschüre 92, Siemens ZPL 1 QE: Zentralabteilung Produktion und Logistik, Qualitätsaufgaben und Entwicklungsmanagement, 1991.

[Lie04] Liepert, B. Industrierobotik, Quo vadis? In *Proc. of Robotik.* Berlin, S. 1-2, 17.-18. Juni 2004.

[Lin98] Lindemann, U.; Birkhofer, H. Empirical Design Research - its contribution to a Universal Design Theory. In *Universal Design Theory, Grabowski, H. (Hrsg.).* Shaker Verlag, Aachen, ISBN 3-8265-4265-7, S. 291-308, 1998.

[Lin01] Lindemann, U. Methoden in der Produktentwicklung. *Konstruktion*, (1/2):3, 2001.

[Lin05] Lindemann, U. *Methodische Entwicklung technischer Produkte - Methoden flexibel und situationsgerecht anwenden.* Springer-Verlag, 2005.

[Lip00] Lippardt, S. *Gezielte Förderung der Kreativität durch bildliche Produktmodelle.* VDI Verlag, Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, Düsseldorf, ISBN 3-18-332501-2, 2000.

[Lun06] Lunze, J. *Regelungstechnik 1 - Systemtheoretische Grundlagen, Analyse und Entwurf einschleifiger Regelungen.* Springer Verlag, 2006.

[Map10] Maplesoft. *Homepage des Herstellers des CAS Maple.* http://www.maplesoft.com/, *zuletzt Aufgerufen am 30.1.10*, 2010.

[May90] Mayer, R.E.; Gallini, J.K. When is a illustration worth ten thousand words? *Journal of Educational Psychology*, 82(4):715–726, 1990.

[Mei09] Meinzer, A. Entwicklung eines Konzepts zur Modellierung von Kosten im Rahmen der objektorientierten Beschreibung von Parallelrobotern. unveröffentlichte Projektarbeit, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 2009.

[Mer05] Merlet, J.P. The Necessity of Optimal Design for Parallel Machines and a Possible Certified Methodology. In *The 2nd International Colloquium of the Collaborative Research Center 562.* Shaker Verlag, Braunschweig, ISBN 3-8322-3866-2, S. 7-20, 10.-11. Mai 2005.

[Mer06] Merlet, J.-P. *Parallel Robots.* Springer Verlag, Dordrecht, ISBN 1-4020-4132-2, 2006.

[Mey09] Meyer, C. Systematische Analyse und Beurteilung von Vorgehensweisen zur Entwicklung von Baukastenstrategien. unveröffentlichte Studienarbeit, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 2009.

[Mül08] Müller, P.; Stark, R. Detecting and Structuring Requirements for the Development of Product-Service Systems. In *19. Symposium Design for X*. Neukirchen, ISBN 978-3-9808539-6-5, S. 1-10, 9.-10. Oktober 2008.

[Mot08] Motte, D. A Review of the Fundamentals of Systematic Engineering Design Process Models. In *10th International Design Conference - Design 2008*. Dubrovnik/ Kroatien, ISBN 978-953-6313-93-8, S. 199-210, 19.-22. Mai 2008.

[Nas53] Nasvytis, A. *Die Gesetzmäßigkeiten kombinatorischer Technik.* Springer Verlag, Berlin, 1953.

[Obe06] Oberender, C. *Die Nutzungsphase und ihre Bedeutung für die Entwicklung umweltgerechter Produkte.* VDI-Verlag GmbH, Dissertation, Fachgebiet Produktentwicklung und Maschinenelemente der Technischen Universität Darmstadt, Düsseldorf, ISBN 3-18-338501-5, 2006.

[Obj09] Object Management Group. *Homepage der OMG Systems Modeling Language*. http://www.omgsysml.org/, *zuletzt Aufgerufen am 27.7.09*, 2009.

[Ort98] Ort, A. *Entwicklungsbegleitende Kalkulation mit Teilebibliotheken.* Papierflieger, Dissertation, Technische Universität Clausthal, Clausthal-Zellerfeld, ISBN 3-89720-372-3, 1998.

[Otr05] Otremba, R. *Systematische Entwicklung von Gelenken für Parallelroboter.* Logos Verlag, Dissertation, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, Berlin, ISBN 3-8325-0811-2, 2005.

[Ott97] Otto, K.N.; Ahrens, G. Eine Methode zur Definition technischer Produktanforderungen. *Konstruktion*, 49(11-12):19–25, 1997.

[Pah07] Pahl, G.; Beitz, W.; Feldhusen, J.; Grote, K.-H. *Konstruktionslehre - Grundlagen erfolgreicher Produktentwicklung, Methoden und Anwendung.* Springer-Verlag, 2007.

[Pav06] Pavlovic, N.; Keimer, R.; Franke, H.-J. Systematic Developement of Adaptronic Joints for Parallel Kinematic Structures. In *Proc. of the 8th International IFAC Symposium on Robot Control - SYROCO '06.* Bologna/Italien, S. 74, 6.-8. September 2006.

[Pav08] Pavlovic, N.; Keimer, R.; Franke, H.-J. Improvement of Overall Performance of Parallel Robots Using Joints with Integrated Piezo-Actuators. In *Proc. of the 11th International Conference on New Actuators - ACTUATOR 2008*. Bremen, S. 121-124, 9.-11. Juni 2008.

[Phi04] Philippsen, H.-W. *Einstieg in die Regelungstechnik - Vorgehensmodell für den praktischen Reglerentwurf.* Fachbuchverlag Leipzig im Carl Hanser Verlag, München, ISBN 3-446-22377-0, 2004.

[Pik06] Pikalek, C. Beschreibung der Metriken zur Qualitätssicherung. *www.sophist.de*, page 6, 2006.

[Pim94] Pimmler, T.U.; Eppinger, S.D. Integration analysis of product decompositions. In *6th International Conference on Design Theory and Methodology*. Minneapolis, ISBN 079-1812-82-0, S. 343–351, 1994.

[Pis00] Pisla, D.; Kerle, H. Development of Dynamic Models for Parallel Robots with Equivalent Lumped Masses. In *Proc. of 6th Int. Conference on Methods and Models in Automation and Robotics*. Miedzyzdroje/ Polen, S. 637-642, 2000.

[Poh05] Pohl, K.; Böckle, G.; van der Linden, F. *Software product line engineering: foundations, principles and techniques.* Springer-Verlag, Berlin, ISBN 978-3-540-24372-4, 2005.

[Pri97a] Pritschow, G.; Wurst, K.-H. LINAPOD - Ein Baukastensystem für Stabkinematiken. *wt -Werkstatttechnik*, 87(9):437–440, 1997.

[Pri97b] Pritschow, G.; Wurst, K.-H. Zur Gestaltungs- und Konstruktionssystematik von Maschinen mit Stabkinematiken. *wt -Produkt und Management*, 87(1/2):46–51, 1997.

[Pri00] Pritschow, G. Parallel Kinematic Machines (PKM) - Limitations and New Solutions. *CIRP Annals-Manufacturing Technology*, 49(1):275–295, 2000.

[Pul04] Pulm, U. *Eine systemtheoretische Betrachtung der Produktentwicklung.* Verlag Dr. Hut, Dissertation, Lehrstuhl für Produktentwicklung der Technischen Universität München, München, ISBN 3-89963-062-9, 2004.

[Qui00] Quirmbach, O. *Integration der System- und Kostenentwicklung am Beispiel eines Satellitensystems.* Herbert Utz Verlag, Dissertation, Technische Universität München, München, ISBN 3-8316-0019-8, 2000.

[Raa91] Raasch, J. *Systementwicklung mit Strukturierten Methoden: Ein Leitfaden für Praxis und Studium.* Carl Hanser Verlag, München, ISBN 3-446-16147-3, 1991.

[REF85] REFA Verband für Arbeitsstudien und Betriebsorganisation e. V. *Methodenlehre des Arbeitsstudiums : Teil 3 Kostenrechnung.* Carl Hanser Verlag, München, ISBN 3-446-14236-3, 1985.

[Ren07] Renner, I. *Methodische Unterstützung funktionsorientierter Baukastenentwicklung am Beispiel Automobil.* Verlag Dr. Hut, Dissertation, Lehrstuhl für Produktentwicklung der Technischen Universität München, München, ISBN 978-3-89963-567-6, 2007.

[Rie10] Riechelmann, C. Anwendung neuer Entwicklungsmethoden für den Entwurf eines modularen Bootskonzepts für den privaten Einsatz. unveröffentlichte Studienarbeit, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 2010.

[Rob08] Robertson, J.; Robertson, S. *Volere Requirements Ressources.* http://www.volere.co.uk/, *zuletzt Aufgerufen am 20.10.08*, 2008.

[Rot71] Roth, K.; Franke, H.-J.; Simonek, R. Algorithmisches Auswahlverfahren zur Konstruktion mit Katalogen. *Feinwerktechnik*, (8):337–364, 1971.

[Rot72] Roth, K.; Franke, H.-J.; Simonek, R. Aufbau und Verwendung von Katalogen für das methodische Konstruieren. *Konstruktion*, (24):449–458, 1972.

[Rot94] Roth, K. *Konstruieren mit Konstruktionskatalogen, Bd. 1 Konstruktionslehre.* Springer-Verlag, Berlin, ISBN 3-540-57324-0, 1994.

[Rup07] Rupp, C. *Requirements-Engineering und Management - Professionelle, iterative Anforderungsanalyse für die Praxis.* Carl Hanser Verlag, München, ISBN 3-446-40509-7, 2007.

[Sal08] Salustri, F.A.; Eng, N.L.; Weerasinghe, J.S. Visualizing Information in the Early Stages of Engineering Design. *Computer-Aided Design and Applications*, 5(5):697–714, 2008.

[Sch01] Schneider, M. *Methodeneinsatz in der Produktentwicklungs-Praxis - Empirische Analyse, Modellierung, Optimierung und Erprobung.* VDI-Verlag GmbH, Dissertation, Fachgebiet Produktentwicklung und Maschinenelemente der Technischen Universität Darmstadt, Düsseldorf, ISBN 3-18-334601-X, 2001.

[Sch02] Schichtel, M. *Produktdatenmodellierung in der Praxis.* Carl Hanser Verlag, München, ISBN 3-446-21857-2, 2002.

[Sch05] Schäppi, B.; Radermacher, F.-J.; Andreasen, M. M.; Kirchgeorg, M. *Handbuch Produktentwicklung.* Carl Hanser Verlag, München, ISBN 3-446-22838-1, 2005.

[Sch09] Schmitt, J.; Stechert, C.; Raatz, A.; Hesselbach, J., Franke, H.-J.; Vietor, T. Reconfigurable Parallel Kinematic Structures for Spreading Requirements. In *Proc. of IASTED - Robotics and Applications.* Cambridge/USA, 2.-4. November 2009.

[Sek05] Sekolec, R. *Produktstrukturierung als Instrument des Variantenmanagements in der methodischen Entwicklung modularer Produktfamilien.* VDI Verlag, Dissertation, Eidgenössisch Technische Hochschule Zürich, Düsseldorf, ISBN 3-18-317216-X, 2005.

[Set04] Setchi, R.-M.; Lagos, N. Reconfigurability and Reconfigurable Manufacturing Systems - State-of-the-art Review. In *2nd IEEE International Conference on Industrial Informatics.* Berlin, ISBN 0-7803-8513-6, S. 529-535, 24.-26. Juni 2004.

[Sic08] Siciliano, B.; Khatib, O. (Hrsg.). *Handbook of Robotics.* Springer, Berlin, ISBN 978-3-540-23957-4, 2008.

[Sie10] Siemens - UGS. *Homepage des Herstellers von Solid Edge.* http://www.solid-system-team.de/, *zuletzt Aufgerufen am 30.1.10*, 2010.

[Sim74] Simonek, R. *Ein Beitrag zur Ermittlung der speziellen Funktionsstruktur in der Konstruktion.* Dissertation, Institut für Konstruktionslehre, Maschinen- und Feinwerkelemente der Technischen Universität Braunschweig, 1974.

[Ste93] Steinmetz, O. *Die Strategie der integrierten Produktentwicklung: Softwaretechnik und Organisationsmethoden zur Optimierung der Produktentwicklung im Unternehmen.* Vieweg Verlag, Wiesbaden, ISBN 3-528-05328-3, 1993.

[Ste06a] Stechert, C.; Pavlovic, N.; Franke, H.-J. Demand-Driven Development of Parallel Robots with Adaptronic Components by a Modular System. In *IEEE International Conference on Robotics and Biomimetics - ROBIO 06.* Kunming/ China, ISBN 1-4244-0571-8, S. 666-671, 17.-20. Dezember 2006.

[Ste06b] Stechert, C.; Wrege, C.; Franke, H.-J. Task-Based Modular Configurations for Hybrid and Redundant Parallel Robots. In *8th International IFAC Symposium on Robot Control*. Bologna, 6.-8. September 2006.

[Ste07a] Stechert, C.; Alexandrescu, I.; Franke, H.-J. Modelling of Inter-Model Relations for a Customer Oriented Development of Complex Products. In *The 16th International Conference on Engineering Design ICED 07*. Paris/Frankreich, ISBN 1-904670-02-4, 28.-30. August 2007.

[Ste07b] Stechert, C.; Pavlovic, N.; Franke, H.-J. Parallel Robots with Adaptronic Components - Design Through Different Knowledge Domains. In *12th IFToMM World Congress*. Besancon/ Frankreich, 17.-21. Juni 2007.

[Ste08a] Stechert, C.; Bauer, S.; Franke, H.-J.; Meerkamm, H. Requirements Management in Early Stages of Mechatronic Design by Visualisation of Interdependencies. In *The 10th International Design Conference, DESIGN 2008*. Dubrovnik/Kroatien, ISBN 953-6313-90-1, Seiten 501-508, 19.-22. Mai 2008.

[Ste08b] Stechert, C.; Franke, H.-J. Managing Requirements as the Core of Multi-Disciplinary Product Development. In *The 18th CIRP Design Conference - Design Synthesis*. Enschede/Niederlande, 7.-9. April 2008.

[Ste09a] Stechert, C.; Franke, H.-J. Managing Requirements as the Core of Multi-Disciplinary Product Development. *CIRP Journal of Manufacturing Science and Technology*, 1(3):153–158, 2009.

[Ste09b] Stechert, C.; Franke, H.-J. Requirements Models for Collaborative Product Development. In *The 19th CIRP Design Conference*. Cranfield/UK, S. 24-31, 30.-31. March 2009.

[Ste09c] Stechert, C.; Franke, H.-J. Requirements Models for Modular Products. In *International Conference on Research into Design*. Bangalore/Indien, S. 411-418, 7.-9. Januar 2009.

[Str08] Straube, D.; Ziebart, J.-R.; Triltsch, U.; Franke, H.-J.; Büttgenbach, S. Computer Aided Development Environment for Active Microsystems. In *The 1st Symposium in Multidisciplinary Studies of Design in Mechanical Engineering*. Bertinoro/Italien, ISBN 88-901080-3-7, Seiten 15-16, 26.-28. Juni 2008.

[Suh90] Suh, N.P. *The Principles of Design*. Oxford University Press, 1990.

[Sur09] Suray, A. Entwicklung und Modellierung eines Baukastensystems für die Gestelle von Parallelrobotern. unveröffentlichte Diplomarbeit, Institut für Konstruktionstechnik der Technischen Universität Braunschweig, 2009.

[Sys10] SysML-Forum. *Vergleich von SysML-Tools des SysML Forums*. http://www.sysmlforum.com/tools.htm, *zuletzt Aufgerufen am 26.1.10*, 2010.

[Tav08] Tavassoli, D. Ten Steps to Better Requirements Management. *Telelogic White Paper*, (7):7, 2008.

[Tre98] Treib, T.; Meier, P.; Hebsacker, M. Wachstumsgesetzmäßigkeiten und Einsatzpotentiale parallel-kinematischer Manipulatoren. In *Proc. of VDI-Fachtagung - Neue Maschinenkonzepte mit parallelen Strukturen für Handhabung und Produktion*. Braunschweig, S. 81-94, 1998.

[Tsc00] Tschakarow, S.; Tschakarow, R.-K. Modulare Roboter - ein neuer Weg in der Robotertechnik. In *Proc. of Robotik*. Berlin, S. 265-271, 2000.

[Ull92] Ullman, D.G. *The Mechanical Design Process*. McGraw-Hill, 1992.

[van08] van Beek, T.J.; Tomiyama, T. Requirements for Complex Systems Modelling. In *The 18th CIRP Design Conference - Design Synthesis*. Enschede/Niederlande, ISBN 978-90-365-2634-0, S. 32, 7.-9. April 2008.

[VDI80] VDI-Richtlinie 2220. *Produktplanung; Ablauf, Begriffe und Organisation*. VDI-Gesellschaft Entwicklung Konstruktion Vertrieb, Düsseldorf, 1980.

[VDI93] VDI-Richtlinie 2221. *Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte*. VDI-Gesellschaft Entwicklung Konstruktion Vertrieb, Düsseldorf, 1993.

[VDI97] VDI-Richtlinie 2222, Blatt 1. *Konstruktionsmethodik - Methodisches Entwickeln von Lösungsprinzipien*. VDI-Gesellschaft Entwicklung Konstruktion Vertrieb, Düsseldorf, 1997.

[VDI98] VDI-Richtlinie 2225 Blatt 1. *Technisch-Wirtschaftliches Konstruieren - Vereinfachte Kostenermittlung*. VDI-Gesellschaft Entwicklung, Konstruktion, Vertrieb, Düsseldorf, 1998.

[VDI01] VDI-Richtlinie 2519 Blatt 1. *Vorgehensweise bei der Erstellung von Lasten-/Pflichtenheften*. VDI-Gesellschaft Fördertechnik Materialfluss Logistik, Düsseldorf, 2001.

[VDI04] VDI-Richtlinie 2206. *Entwicklungsmethodik für mechatronische Systeme.* VDI-Gesellschaft Entwicklung Konstruktion Vertrieb, Düsseldorf, 2004.

[VDI07] VDI-Richtlinie 2156. *Einfache räumliche Kurbelgetriebe; Systematik und Begriffsbestimmungen.* VDI-Gesellschaft Entwicklung, Konstruktion, Vertrieb, Düsseldorf, 2007.

[VDI08] VDI/VDE-Richtlinie 3694. *Lastenheft/Pflichtenheft für den Einsatz von Automatisierungssystemen.* VDI/VDE-Gesellschaft Mess- und Automatisierungstechnik, Düsseldorf, 2008.

[Vie09] Vietor, T. Konstruktionsmethodik – Ein Schlüssel für neue Fahrzeugkonzepte? In *Antrittsvorlesung an der technischen Universität Braunschweig.* 16. Dezember 2009.

[Wah03] Wahl, F.M.; Krefft, M. (Hrsg.). *Sonderforschungsbereich 562–Antrag auf Finanzierung und Ergebnisbericht.* SFB 562, Braunschweig, 2003.

[Wah06] Wahl, F.M.; Last, P. (Hrsg.). *Sonderforschungsbereich 562–Antrag auf Finanzierung und Ergebnisbericht.* SFB 562, Braunschweig, 2006.

[Web05] Weber, C. CPM/PDD - An Extended Theoretical Approach to Modelling Products and Product Development Processes. In *The 2. German-Israeli Symposium for Design and Manufacture „Advances in Methods and Systems for Products and Processes", Bley, H.; Jansen, H.; Krause, F.-L.; Shpitalni, M. (Hrsg.).* Berlin, ISBN 3-8167-6874-1, S. 159-179, 7.-8. Juli 2005.

[Wei91] Weidenmann, B. *Lernen mit Bildmedien. Psychologische und didaktische Grundlagen.* Verlag Beltz, Basel, ISBN 3-407-36001-0, 1991.

[Wei06] Weilkiens, T. *Systems Engineering mit SysML/UML - Modellierung, Analyse, Design.* dpunkt.verlag GmbH, 2006.

[Wol94] Wolf, W. *Modern VLSI Design - A Systems Approach.* Prentice-Hall, 1994.

[Wol05] Wolf, H.; Roock, S.; Lippert, M. *eXtreme Programming - Eine Einführung mit Empfehlungen und Erfahrungen aus der Praxis.* dpunkt.verlag GmbH, Heidelberg, ISBN 3-89864-339-5, 2005.

[Wur02] Wurst, K.-H.; Peting, U. PKM Concept for Reconfigurable Machine Tools. In *4th Chemnitz Parallel Kinematic Seminar.* Verlag wissenschaftliche Scripten, Zwickau, ISBN 978-3-928921-76-3, S. 683-696, 2002.

[Yan00] Yang, G.; Chen, I.-M. Task-Based Optimization of Modular Robot Configurations: Minimized Degree-of-Freedom Approach. *Journal of Mechanism and Machine Theory*, 35(4):517–540, 2000.

[Yoo02] Yoon, J.; Ryu, J. Reconfigurability of a Parallel Manipulator: A Case Study. In *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*. Quebec/ Kanada, S. 94-97, 2002.

[Zan99] Zanker, W. *Situative Anpassung und Neukombination von Entwicklungsmethoden*. Shaker Verlag, Dissertation, Lehrstuhl für Produktentwicklung der Technischen Universität München, Aachen, ISBN 3-8265-6558-4, 1999.

[Zau09] Zaur Nasibov. *The „tree swing picture“ of Requirements Management*. http://www.znasibov.info/data/img/life-in-pictures-swing.png, *zuletzt Aufgerufen am 07.9.09*, 2009.

[Zwi66] Zwicky, F. *Entdecken, Erfinden, Forschen im morphologischen Weltbild*. Droemersche Verlagsanstalt Th. Knaur, München, 1966.

Abbildungsverzeichnis

1.1 Herkunft und strukturierte Verwendung von Anforderungen. . . . 3
1.2 Grafischer Überblick über den Aufbau dieser Arbeit. 4

2.1 Probleme der Produktentwicklung. 6
2.2 Bausteine der Produktentwicklung. 7
2.3 Baustein „Systemidee beschreiben“. 9
2.4 Baustein „Situation analysieren“. 10
2.5 Baustein „Ziele formulieren“. 11
2.6 Baustein „Projekt planen und überwachen“. 12
2.7 Baustein „Systemkontext beschreiben“. 13
2.8 Baustein „Fachwissen und Entscheidungen dokumentieren“. 14
2.9 Baustein „Anforderungen erfassen“. 14
2.10 Baustein „Anforderungen bereitstellen“. 16
2.11 Baustein „Anforderungen verarbeiten“. 17
2.12 Die verteilte Produktentwicklung als Teamsport. 18
2.13 Der Nutzen von Formalisierung. 24
2.14 Stoßrichtungen der Baukastenentwicklung. 29

3.1 Die drei Aspekte der Modellbildung. 38
3.2 Verwendung von Modellen im Produktentwicklungsprozess. 41
3.3 Beeinflussungsmatrix der Partialmodellbeziehungen. 44
3.4 Der Aufbau der SysML-Diagramme. 47
3.5 Unterschiede verschiedener Modellversionen. 49
3.6 Übersicht zur Unterteilung von Beziehungsarten. 53
3.7 Biegesteifigkeit und Masse eines Rechteckträgers. 57
3.8 Mechanismen beim Umgang mit Zielkonflikten. 63

4.1 Partialmodelle zur Produktmodellierung. 71
4.2 Package-Browsers in Artisan Studio. 72
4.3 Diagramm der Systemidee „Flexibilität“. 74
4.4 Ziele der Strategie „Baukastenentwicklung“. 75
4.5 Phasen des Produktlebenslaufs. 76
4.6 Anwendungsfall Getriebemontage. 77

4.7 Produktumgebung Robotersystem. 79
4.8 Hierarchische Stakeholderstruktur. 80
4.9 Ausschnitt eines Anforderungsdiagramms. 84
4.10 Parametrikdiagramm für Anforderungsbeziehungen. 85
4.11 Anforderungshierarchie im Packagebrowser. 86
4.12 Hauptbaugruppen des Robotersystems. 87
4.13 Hauptbaugruppe „Kinematische Struktur“ des Robotersystems. . . 88
4.14 Konstruktionskatalog Testfallklassen. 91
4.15 Anforderungsdiagramm für Testfälle, -kriterien und -werkzeuge. . 96
4.16 Excelformular zur Modellauswertung. 98
4.17 Stakeholder-Anwendungsfälle-Matrix für Parallelroboter. 99
4.18 Anforderungen-Anwendungsfälle-Matrix für Parallelroboter. . . . 100
4.19 Anforderungen-Komponenten-Matrix für Parallelroboter. 102
4.20 Testkriterien-Testfälle-Matrix für Parallelroboter. 104
4.21 Anforderungen-Anforderungen-Matrix für Parallelroboter. 105
4.22 Anforderungsstatus im Zeitverlauf. 109
4.23 Zeitlicher Verlauf der Anforderungsklärung. 110
4.24 Priorisierung/ Quantifizierung von Anforderungen im Zeitverlauf. 110
4.25 Erfüllbare/ sichere Anforderungen im Zeitverlauf. 111
4.26 Beziehungen und Anforderungsqualität im Zeitverlauf. 112
4.27 Ablaufplan Baukastenentwicklung. 113

5.1 Foto des Modellautos. 119
5.2 Auswertung der Fragebögen im Wintersemester 2008/2009. 122
5.3 Ein Anforderungsdiagramm. 123
5.4 Struktur des SFB 562. 126
5.5 Funktionsplan für Industrieroboter. 126
5.6 Unterschiedlicher Aufbau eines Bewegungswandlers. 127
5.7 Dokumentation der Systemidee für einen Parallelroboter. 130
5.8 Dokumentation der Ziele für einen Parallelroboter im Modell. . . 131
5.9 Satisfy-Matrix Systemidee-Ziele eines Parallelroboters. 131
5.10 Strukturierung der Aufgaben des Robotersystems. 133
5.11 Fördermittel zur Zuführung bei Parallelrobotern. 134
5.12 Anwendungsfälle-Stakeholder-Matrix für Parallelroboter. 135
5.13 Suchergebnisse zur Anforderungsspreizung. 136
5.14 Arbeitsblatt zur Zielkonfliktanalyse. 138
5.15 Anforderungsdiagramm zur Zielkonfliktanalyse. 139
5.16 Strukturschema Baukastenkonzept. 141
5.17 Diagramm zur Darstellung der Intermodellbeziehungen. 142
5.18 Rechnerunterstützte Konstruktionskataloge. 143
5.19 Konzept für Robotergestelle. 144
5.20 Funktionalitäten Entwicklungsumgebung SFB562. 145
5.21 Testfall zur Arbeitsraumanalyse. 147

5.22 Strukturierte Anwendungsfälle und Anforderungsspreizung. 150
5.23 Hierarchisches Zielsystem. 151
5.24 Beziehungen Anforderungen und Komponenten. 152
5.25 Einflussanalyse eines Rekonfigurationsparameters. 153
5.26 Rekonfigurationsmodule für Parallelroboter. 154
5.27 Ebene kinematische Struktur. 156
5.28 Einfluss der Rekonfigurationsparameter. 157
5.29 Beziehungssystem zur Rekonfigurationsparameteridentifikation. . . 158
5.30 Anwendungsfälle vor und nach Rekonfiguration. 159

C.1 Fragebogen Wintersemester 2008/2009. 220

Tabellenverzeichnis

2.1 Die wichtigsten Unterscheidungsmerkmale für Baukastensysteme. 27

3.1 Strukturierungsmerkmale für unternehmerische Anforderungen. . 51
3.2 Technische Anforderungen als Strukturierungsmerkmale. 52

4.1 Gliederungsmerkmale und ihre Ausprägungen für Testfälle. 90
4.2 Schematische Darstellung einer „Support"-Matrix. 106

5.1 Bei einer Rekonfiguration zu beeinflussende Parameter. 155

A.1 Berücksichtigung der Bausteine der Produktentwicklung. 204
A.1 *Fortsetzung* 205

B.1 Allgemeine Ziele, die von Modellen erfüllt werden sollen. 209
B.1 *Fortsetzung* 210
B.2 Allgemeine Merkmale zur Modellcharakterisierung. 210
B.2 *Fortsetzung* 211
B.3 Gegenüberstellung von Modellierungszielen. 212
B.4 Analyse bekannter Modellierungssprachen. 213

Glossar

«goal»

SFB562 Stereotyp: Übergeordnete strategische Ziele ohne Bindung an eine Umsetzung durch eine bestimmte Produktgruppe (z.B. kurze Zykluszeiten bei der Handhabung eines Objektes). Fasst neue *tag definitions* zusammen und wird auf «requirement» angewendet.

«reqattributes»

SFB562 Stereotyp: Fasst neue *tag definitions* für Anforderungen zusammen und wird auf «requirement» angewendet.

«requirement»

SysML Stereotyp: Anforderung als gewünschte, zu realisierende Produkteigenschaft.

«target»

SFB562 Stereotyp: Ziele, die direkt durch das Produkt erfüllt werden können und die übergeordneten Ziele unterstützen (z.B. hohe Dynamik des Robotersystems). Fasst neue *tag definitions* zusammen und wird auf «requirement» angewendet.

«testcase»

SysML/SFB562 Stereotyp: Beschreibt Testfälle zur Überprüfung des Systems. Erweitert den vorhandenen SysML Stereotypen durch neue *tag definitions* und wird auf «usecase» angewendet.

«testcriterion»

SFB562 Stereotyp: Ein Testkriterium oder Abnahmekriterium beschreibt die Eigenschaften, sie bei einem Test überprüft werden. Fasst neue *tag definitions* für Anforderungen zusammen und wird auf «requirement» angewendet.

«usecase»

SysML Stereotyp: Ein Anwendungsfall ist eine formalisierte Beschreibung einer Handlung innerhalb des Produktlebenslaufs.

Anforderung

Geforderte Eigenschaft eines Produktes bzw. Vorgabe für die zielgerichtete Entwicklung. Bei dem Produkt kann es sich z.B. um einen Gegenstand (z.B. Auto) oder einen Prozess (z.B. Einkaufsabwicklung) handeln.

Anforderungskollektiv

Ein Anforderungskollektiv beinhaltet sämtliche für einen bestimmten Entwicklungsschritt (Konkretisierungsstufe, Komponenten, Varianten) relevanten Anforderungen.

Attribute

Attributes werden im objektorientierten Modell genutzt, um einzelne Eigenschaften eines Objekts darzustellen.

Computer Aided Design (CAD)

Das Computer Aided Design umfasst im Wesentlichen geometrische Modellierer, die das Produkt virtuell im Rechner abbilden. Aus den heute üblichen 3D-Modellen können z.B. 2D-Fertigungszeichnungen abgeleitet werden oder die Daten können als Grundlage für weiterführende Berechnungs- und Simulationswerkzeuge genutzt werden.

Computer Algebra System (CAS)

Bei Computer Algebra Systemen handelt es sich um Mathematiksoftware. Dabei werden die Berechnungen im Wesentlichen algebraisch durchgeführt und nur bei Bedarf werden Werte numerisch bestimmt.

Constraint-based development environment (CBDE)

Bezeichnet eine constraintbasierte Entwicklungsumgebung. Durch die Integration von Constraintsolvern und z.B. CAD-Systemen können während des Entwicklungsprozesses bekannte Bedingungen automatisch gelöst werden. Beispielsweise können, basierend auf Regelwerken zur Druckbehälterauslegung, halbautomatisch Druckbehälter für spezielle Aufgaben ausgelegt und CAD-Modelle erzeugt werden [Bre03].

Computational Fluid Dynamics (CFD)

Beim Computational Fluid Dynamics werden Strömungsprobleme gelöst, indem der zu betrachtende Raum in finite Elemente zerlegt wird und die Strömungsgleichungen innerhalb der Elemente numerisch gelöst werden.

Deutsche Forschungsgemeinschaft (DFG)

Die Deutsche Forschungsgemeinschaft ist die zentrale Selbstverwaltungseinrichtung der Wissenschaft zur Förderung der Forschung an Hochschulen und

öffentlich finanzierten Forschungsinstitutionen in Deutschland. Sie dient der Wissenschaft in allen ihren Zweigen durch die finanzielle Unterstützung von Forschungsvorhaben und durch die Förderung der Zusammenarbeit unter den Forschern [Deu08].

Design Structure Matrix (DSM)

Design Structure Matrices dienen der systematischen Darstellung von Beziehungen zwischen Eigenschaften des gleichen oder unterschiedlicher Partialmodelle eines Produktes. Beispielsweise können Anforderungen auf Funktionen und diese auf Wirkprinzipe o.ä. abgebildet werden.

Finite Elemente Methode (FEM)

Bei der Finiten Elemente Methode werden Probleme der Spannungsberechnung gelöst, indem der zu betrachtende Raum in finite Elemente zerlegt wird und die Gleichungen innerhalb der Elemente numerisch gelöst werden.

Failure Mode and Effects Analysis (FMEA)

Die Fehlermöglichkeits- und -einflussanalyse untersucht ein Produkt oder einen Prozess systematisch auf mögliche Fehler und deren Einfluss auf das Gesamtsystem. Durch einen daraus abgeleiteten Maßnahmenkatalog soll erreicht werden, dass keine Fehler auftreten.

Fault Tree Analysis (FTA)

Durch eine Fehlerbaumanalyse sollen die Ursachen möglicher Fehler identifiziert werden. Durch einen daraus abgeleiteten Maßnahmenkatalog soll erreicht werden, dass keine Fehler auftreten.

Gel-Permeations-Chromatographie (GPC)

Bei der Gel-Permeations-Chromatographie handelt es sich um eine Art der Flüssigchromatographie. Die Trennung der chemischen Stoffe findet hier aufgrund der Größe (genauer: dem hydrodynamischen Volumen) der Moleküle in Lösung statt. Die Säulen sind je nach Anwendung mit unterschiedlichen Stoffen verschiedener Porengrößen befüllt.

House of Qualities (HoQ)

Das House of Qualities des QFD ist eine grafische Repräsentation der Beziehungen zwischen den Kundenwünschen (Kopfspalte) und Produktmerkmalen (Kopfzeile). Das „Dach" wird durch eine Dreiecksmatrix gebildet, die Intramodellbeziehungen zwischen Produktmerkmalen abbildet.

Institute of Electrical and Electronics Engineers (IEEE)

Ein Weltverband von Ingenieuren aus dem Bereichen Elektrotechnik und Informatik mit Sitz in New York. Er ist z.B. Veranstalter von Tagungen und Herausgeber von IEEE Standard (IEEE Std).

IEEE Standard (IEEE Std)

Verschiedene Gremien des Institute of Electrical and Electronics Engineers (IEEE) erarbeiten Richtlinien. Diese IEEE Standards gelten als allgemein anerkannter Stand der Technik und Forschung. Sie sind im Allgemeinen jedoch nicht rechtlich bindend.

Informationstechnologie (IT)

Die Informationstechnologie befasst sich allgemein mit der Informations- und Datenverarbeitung, sowie der dazu benötigten Hard- und Software.

Klasse

Eine Menge von Objekten mit gleichen Merkmalen kann in eine Klasse zusammengefasst werden. Durch Klassifikation können Objekte strukturiert werden. Beispielsweise kann die Menge aller Fahrzeuge in die Klassen Zweiräder, Dreiräder, Vierräder usw. eingeteilt werden, so dass ein Motorrad der ersten Klasse zugezählt wird.

MAPS

Ein auf Matlab basierendes rechnerunterstütztes Werkzeug zur Kinematikanalyse von Parallelstrukturen. Es wurde am IWF der TU Braunschweig entwickelt.

Market-Pull

Wird ein Produkt zur Erfüllung eines bekannten Kundenbedürfnisses eines bestimmten Marktsegments (Zielgruppe) entwickelt, so spricht man von einem Market-Pull.

Module Interface Graph (MIG)

Die vereinfachte Darstellung von Modulen eines Baukastenproukts in Form von einfachen Geometrien in einer Ebene, sowie den Schnittstellen zwischen ihnen als Linien [Ble08a].

Mehrkörpersimulation (MKS)

Die Mehrkörpersimulation simuliert die Kinematik mehrere Körper. Dabei werden Körper zumeist als starre Körper mit Punktmassen modelliert, die durch Freiheitsgradeinschränkungen und ggf. elastische Elemente miteinander verbunden sind.

Objekt

Ein Objekt ist ein Element der realen Welt (z.B. ein Maschinenelement) oder einer abstrakten Darstellung (z.B. eine Funktion). Objekte besitzen Eigenschaften und Operationen.

Partialmodell

Partialmodelle (oder P_n-Modelle) sind durch logisch zusammenhängende Inhalte gekennzeichnet, die mindestens diejenigen Systemeigenschaften der jeweiligen Konkretisierungsstufe darstellen. In ihrer Summe bilden sie das integrierte Produktmodell.

Produktlebenslauf

Der Produktlebenslauf umfasst den Zeitraum vom Start der Produktplanung, über die Produktentwicklung, Herstellung und Nutzung und endet bei der Außerbetriebnahme, Verwertung und Entsorgung.

Produktumgebung

Die Produktumgebung berücksichtigt alle Gegebenheiten, die nicht eigentlich Teil des zu entwickelnden Produktes sind, aber zu irgendeinem Zeitpunkt des Produktlebens mit ihm in eine Beziehung treten.

Product-Service Systems (PSS)

Product-Service Systems sind Bündel von Produkten und Serviceleistungen, die in Kombination angeboten werden, z.B. Mobiltelefon, Netzinfrastruktur und Serviceleitstungen des Telekomunikationsanbieters.

Quality Function Deployment (QFD)

Das Quality Function Deployment ist eine Methode zur Qualitätssicherung. Dabei sollen durch eine systematische Berücksichtigung der Kundenwünsche nur solche Produkte entwickelt werden, die vom Kunden gewünscht werden.

Requirements Management (RM)

Oberbegriff für alle Tätigkeiten zur Verwaltung und Verwendung von Anforderungen.

Rapid Prototyping (RP)

Das Rapid Prototyping bezeichnet Verfahren mit denen im Allgemeinen aus virtuellen Bauteilen automatisiert reale Bauteile erzeugt werden. Diese bieten meist die gleichen Geometrien, wie das spätere Bauteil aber häufig nicht sein Verhalten, z.B. wegen anderer Werkstoffe oder inhomogener Struktur.

Risikoprioritätszahl (RPZ)

Die Risikoprioritätszahl gibt in der FMEA an, wie hoch das Risiko eines möglichen Fehlers eingeschätzt wird. Dabei setzt sie sich multiplikativ aus der Auftretenswahrscheinlichkeit, der Bedeutung und der Entdeckungswahrscheinlichkeit zusammen. Es wird bei einem Wert zwischen 100 und 200 von einem mittleren Risiko ausgegangen. Kleinere Werte bedeuten ein geringeres, größere Werte ein höheres Risiko.

Systems Engineering (SE)

Das Systems Engineering bezeichnet einen integrierten Ansatz zur Systementwicklung. Hier werden explizit Systeme als oberste Entwicklungsebene betrachtet. Das Vorgehen berücksichtigt die systematische Aufteilung in fachspezifisch zu bearbeitende Module und die gezielte Integration der Teillösungen zu einem Gesamtsystem.

Sonderforschungsbereich (SFB)

Sonderforschungsbereiche sind langfristig angelegte Forschungseinrichtungen der Hochschulen, in denen Wissenschaftler im Rahmen eines fächerübergreifenden Forschungsprogramms zusammenarbeiten. Sie werden durch die Deutsche Forschungsgemeinschaft (DFG) gefördert.

Spezielle Funktionsstruktur (SFS)

Die Spezielle Funktionsstruktur bildet ein Konzept ab, in welchem Eingangsgrößen durch physikalische Effekte verknüpft werden, bis die gewünschten Ausgangsgrößen vorliegen [Sim74].

Situation

Die Situation beschreibt die Umstände, die auf ein Objekt bezogen sind. Diese können z.B. zeitlicher, räumlicher oder persönlicher Natur sein.

Start Of Production (SOP)

Bezeichnet den Serienanlauf eines Produkts, d.h. den Zeitpunkt zu dem die Herstellung des Produkts beginnen soll.

Stakeholder

Als Stakeholder werden natürliche, juristische und abstrakte Personen bezeichnet, die direkten oder indirekten Einfluss auf Anforderungen haben. D.h. ein Stakeholder kann auch für eine ganze Gruppe von Personen stehen (z.B. Anwender) oder eine natürliche Person kann verschiedene Stakeholderrollen in sich vereinen (z.B. Anwender und Einkäufer).

Stereotyp

In der objektorientierten Modellierung werden verschiedene Merkmale in Stereotypen zusammengefasst, die dann auf bestimmte Klassen angewendet werden können.

Systems Modelling Language (SysML)

Die Systems Modelling Language ist eine auf der UML basierende Beschreibungssprache zur abstrakten Modellierung von Systemen [Obj09].

System

Ein System umschließt durch seine Systemgrenze Elemente, die miteinander in Beziehung stehen, und grenzt sie dadurch von der Umgebung ab. Das System ist durch Ein- und Ausgangsgrößen mit seiner Umgebung verbunden.

Systemidee

Die Systemidee beschreibt in kurzer und knapper Form den Wesenskern des Systems. Hier werden also diejenigen Ziele definiert, die einen übergeordneten Charakter besitzen und prinzipiell durch unterschiedliche Produkte lösbar sind. Sie werden z.B. ausgehend vom Kunden als Bedarf formuliert (vgl. *market-pull*). Oder es wird aus einem vorhandenen Produkt die Systemidee abgeleitet, die nun als Verkaufsargument z.B. dem Marketing zur Verfügung steht (vgl. *technology-push*).

Systemkontext

Im Systemkontext werden hier alle konzeptionellen und gestaltenden Maßnahmen und dazugehörigen Modelle zusammengefasst (z.B. Funktionsstrukturen, geometrische Modelle).

tag definition

Eine *tag definition* ist ein in der objektorientierten Modellierungsumgebung festgelegtes Merkmal. Durch *tag definitions* lassen sich die Eigenschaften bestimmter Objekte genau festlegen.

Target Costing (TC)

Das Target Costing ist eine Methode für eine zielgerichtete, kostenorientierte Produktentwicklung, dabei werden Kostenziele vorgegeben und z.B. nach ihrem Anteil am Gesamtnutzen auf die einzelnen Komponenten des Produkts aufgespalten. Dadurch wird frühzeitig ein Kostenbewusstsein beim Entwickler erreicht.

Tool Center Point (TCP)

Der Tool Center Point ist der Zielpunkt für die aufgabenspezifische Manipulation. Meist ist er am Endeffektor des Roboters angeordnet, bspw. der „Greifpunkt" des Greifers. Der Endeffektor bildet das „letzte" Element in der kinematischen Kette.

Technology-Push

Wird durch eine neue Technologie ein neuartiges Produkt ermöglicht, so muss ausgehend von der Technologie ein Marktsegment (eine Zielgruppe) identifiziert werden, in dem durch das neue Produkt ein Kundenbedürfnis erfüllt wird. In Technology-Push Situationen sind die Kundenbedürfnisse häufig nicht im Vorhinein bekannt, sondern werden erst durch das neue Produkt generiert.

Theorie des erfinderischen Problemlösens (TRIZ)

TRIZ ist eine auf den Arbeiten von G.S. Altschuller basierende Methodik zum Lösen von technischen Widersprüchen durch innovative Grundprinzipe [Alt84].

Unified modelling Language (UML)

Die Unified Modelling Language ist eine objektorientierte Beschreibungssprache, die für die Entwicklung komplexer Softwaresysteme entwickelt wurde.

Visual Basic (VB)

Visual Basic ist eine objektorientierte höhere Programmiersprache. Sie ist ein von Microsoft entwickelter Dialekt der in den 1960er Jahren entwickelten Programmiersprache BASIC.

Visual Basic for Applications (VBA)

Eine auf VB basierende Skriptsprache zur Makroprogrammierung innerhalb der Microsoft Office Anwendungen. Sie wird aber auch in einigen von Microsoft unabhängigen kommerziellen Softwareprodukten als Skriptsprache verwendet.

Verein Deutscher Ingenieure (VDI)

Ein deutscher Verein von Ingenieuren verschiedener Fachrichtungen mit Sitz in Düsseldorf. Er ist z.B. Veranstalter von Tagungen und Herausgeber von VDI-RL

VDI-Richtlinie (VDI-RL)

Eine VDI-Richtlinie ist Teil des Regelwerks des Verein Deutscher Ingenieure (VDI) und gilt als allgemein anerkannter Stand der Technik und Forschung. Richtlinien sind im Allgemeinen jedoch nicht rechtlich bindend.

eXtreme Programming (XP)

Das eXtreme Programming ist eine moderne Programmierphilosophie, die verstärkt auf Teamarbeit, Komplexitätsverringerung und Zielbewusstsein setzt [Bec00].

Ziel

Ziele bilden eine z.T. unscharfe Vorstellung dessen ab, was erreicht werden soll. Dabei stammen Ziele von Kunden- und Herstellerseite sowie (z.B. gesetzlichen oder technologischen) Randbedingungen.

Zielkonflikt

Ein Zielkonflikt liegt dann vor, falls die Verbesserung der Erfüllung eines Zieles nur durch eine gleichzeitige Verschlechterung der Erfüllung eines weiteren Zieles erreicht werden kann.

Formelzeichen

a_{max}	Max. konst. Beschleunigung entlang Trajektorie	m/s^2
a_{TCP}	Beschleunigung am TCP	m/s^2
$a(t)$	Beschleunigung abhängig von der Zeit t	m/s^2
b	Breite Rechteckträger	m
b	Abstand Gestellanbindungspunkte **RPRPR**	m
d	Innendurchmesser Spannsatz	mm
g	Erdbeschleunigung	m/s^2
h	Höhe Rechteckträger	m
l	Länge Rechteckträger	m
l_1, l_2	Länge angetriebene Glieder **RPRPR**	m
l_{AR}	Arbeitsraumlänge	m
l_{max}	Max. Länge einer Trajektorie	m
m	Masse	kg
v_{max}	Max. Geschwindigkeit entlang Trajektorie	m/s
$v(t)$	Geschwindigkeit abhängig von der Zeit t	m/s
x, y, z	Variablen	
x_p, y_p	Koordinaten TCP **RPRPR**	
$x(t)$	Weg abhängig von der Zeit t	m
A, B, B'	Gestellanbindungspunkte **RPRPR**	
A_{AQ}	Anforderungsaufbereitungsqualität	[-]
A_{BQ}	Anforderungsbeziehungssqualität	[-]
$B_{Ausschluss}$	Anzahl ausschließender Bez. zwischen Anf.	[-]
$B_{Konkurrenz}$	Anzahl konkurrierender Bez. zwischen Anf.	[-]
$B_{Unterstützung}$	Anzahl unterstützender Bez. zwischen Anf.	[-]
D	Außendurchmesser Spannsatz	mm
E	Elastizitätsmodul	N/mm^2
$F(x)$	Federkraft abhängig vom Weg x	N
α	Überstreichbarer Winkel eines Drehgelenks	°
α	Gestellanbindungsebenenwinkel **RPRPR**	°
δ	Biegesteifigkeit Rechteckträger	Nmm^2
δ	Dämpfungskoeffizient	s^{-1}
ρ	Dichte	g/cm^3
ω	Kreisfrequenz	s^{-1}

Anhang A

Anhang

A.1 Zusammenfassung der Ansätze zur Produktentwicklung

Die folgende Tabelle A.1 zeigt eine systematische Aufstellung verschiedener Ansätze zur Produktentwicklung. Es wurden Ansätze aus den verschiedenen technischen Disziplinen (Maschinenbau, Elektrotechnik, Informationstechnologie) untersucht, die bei der Entwicklung vieler aktueller Produkte von großer Bedeutung sind. Dabei wurden frühe Ansätze (z.B. die „Erfindungslehre“ von *Kesselring*), etablierte Vorgehensweisen (z.B. VDI-RL-2221) und aktuelle Forschungsarbeiten (z.B. Anforderungskonfigurator für spaltprofilierte Bleche [Cha08]) berücksichtigt.

Tabelle A.1: *Zusammenfassung der Ansätze verschiedener Autoren bzgl. der Berücksichtigung der einzelnen Bausteine der Produktentwicklung (○= nicht berücksichtigt, ◑= berücksichtigt, ●= berücksichtigt und methodisch unterstützt, MB = Maschinenbau, ET = Elektrotechnik, IT = Informationstechnologie, SE = Systems Engineering).*

Autor	Jahr	Sichtweise	Systemidee	Situation	Ziele	Projekt	Systemkontext	Dokumentation	Anf. erfassen	Anf. bereitstellen	Anf. aufbereiten
Kesselring, [Kes54]	1954	MB	◑	◑	○	○	◑	◑	○	○	○
Zwicky, [Zwi66]	1966	MB	○	◑	◑	○	◑	◑	◑	○	○
Franke, [Fra76]	1976	MB	○	◑	○	○	◑	◑	●	◑	◑
VDI-RL 2220, [VDI80]	1980	MB	○	◑	◑	●	○	◑	◑	○	○
VDI-RL 2222, [VDI97]	1980	MB	○	◑	◑	○	○	○	◑	○	◑
REFA, [REF85]	1985	MB	○	◑	◑	●	○	○	○	○	○
Böhm, [Boe88]	1988	IT	○	◑	◑	◑	●	◑	◑	◑	◑
Gajski, [Gaj88]	1988	ET	○	○	◑	○	●	○	◑	◑	○
Wasserfallmodell, [Raa91], [Boe88]	1988	IT	○	○	◑	○	●	○	◑	◑	◑
Suh, [Suh90]	1990	MB	○	○	◑	○	◑	◑	◑	◑	◑
Löhnert, [Löh91]	1991	MB	○	◑	○	○	○	◑	●	◑	◑
Prototyping, z.B. [Raa91]	1991	IT	○	○	○	○	◑	○	◑	○	◑
Hubka / Eder, [Hub92]	1992	MB	○	◑	○	○	●	◑	◑	◑	◑
Ullman, [Ull92]	1992	MB	○	●	◑	●	●	◑	●	◑	◑
Breiing / Flemming [Bre93]	1993	MB	○	◑	○	◑	◑	◑	◑	◑	◑
Kläger, [Klä93]	1993	MB	○	◑	◑	○	◑	◑	◑	◑	◑
VDI-RL 2221, [VDI93]	1993	MB	○	◑	○	○	◑	◑	◑	◑	◑
Koller, [Kol94]	1994	MB	○	◑	●	◑	●	◑	◑	◑	◑
Roth, [Rot94]	1994	MB	○	◑	●	◑	●	◑	●	◑	◑
W. Wolf, [Wol94]	1994	ET	○	○	◑	○	◑	○	◑	◑	○
Kickermann, [Kic95]	1995	MB	○	○	○	○	◑	◑	◑	●	●
Humpert, [Hum95]	1995	MB	○	○	○	◑	○	◑	◑	●	●
Danner, [Dan96]	1996	MB	○	◑	◑	○	○	◑	◑	◑	◑
Kruse, [Kru96]	1996	MB	○	○	◑	○	◑	◑	◑	●	◑
Wertanalyse, [DIN02]	1996	MB	○	◑	◑	◑	●	◑	◑	○	○
Otto, [Ott97]	1997	MB	○	○	○	○	◑	○	◑	◑	◑
IEEE Std 1233, [IEE98a]	1998	SE	○	○	○	○	○	◑	◑	◑	◑
IEEE Std 830, [IEE98b]	1998	IT	○	○	○	○	○	◑	◑	◑	◑
Konhäuser, [Kon98]	1998	ET	○	○	○	○	●	○	○	○	○
Eiletz, [Eil99]	1999	MB	○	○	●	○	◑	○	○	○	◑
Beck, [Bec00]	2000	IT	◑	◑	◑	◑	◑	○	◑	◑	○

Tabelle A.1: ***(Fortsetzung)*** *Zusammenfassung der Ansätze verschiedener Autoren bzgl. der Berücksichtigung der einzelnen Bausteine der Produktentwicklung.*

Autor	Jahr	Sichtweise	Systemidee	Situation	Ziele	Projekt	Systemkontext	Dokumentation	Anf. erfassen	Anf. bereitstellen	Anf. aufbereiten
Ahrends, [Ahr00]	2000	MB	○	◑	◑	○	○	○	◑	◑	◑
Krusche, [Kru00]	2000	MB	○	○	○	○	○	○	◑	◑	◑
VDI-RL 2519, [VDI01]	2001	MB	○	◑	◑	○	○	●	◑	◑	○
Daenzer, [Dae02]	2002	SE	○	●	◑	◑	◑	◑	●	◑	◑
Philippsen, [Phi04]	2004	ET	○	○	◑	◑	●	◑	◑	○	○
VDI-RL 2206, [VDI04]	2004	SE	○	◑	◑	○	◑	○	○	◑	○
V-Modell XT[Bun04]	2004	IT	○	○	◑	◑	◑	◑	◑	◑	○
Andreasen, [Sch05]	2005	MB	○	◑	○	◑	◑	○	◑	◑	○
DIN EN ISO 9000, [DIN05]	2005	SE	○	◑	◑	◑	○	◑	◑	◑	◑
H. Wolf, [Wol05]	2005	IT	◑	◑	◑	◑	◑	◑	◑	◑	○
Lindemann, [Lin05]	2005	MB	○	◑	●	○	●	◑	●	◑	◑
Weber, [Web05]	2005	MB	○	○	◑	○	◑	○	○	◑	◑
Lunze, [Lun06]	2006	ET	○	◑	◑	○	●	○	◑	○	○
Czichos, [Czi06]	2006	SE	○	○	○	○	◑	○	○	◑	○
Frank, [Fra06]	2006	SE	○	●	●	○	●	◑	●	◑	◑
Jung, [Jun06]	2006	MB	○	◑	○	○	◑	○	◑	◑	●
Merlet / Gosselin, [Sic08], [Mer06]	2006	MB	○	○	○	○	●	○	○	◑	◑
UfAB IV, [Koo06]	2006	IT	○	○	○	◑	○	◑	◑	◑	○
Weilkiens, [Wei06]	2006	SE	◑	●	●	○	◑	◑	●	●	○
Ehrlenspiel, [Ehr07]	2007	MB	○	◑	◑	◑	●	◑	●	◑	◑
Pahl / Beitz, [Pah07]	2007	MB	○	◑	◑	○	●	◑	●	◑	◑
Rupp, [Rup07]	2007	IT	○	◑	○	◑	○	◑	●	●	◑
Angeles, [Sic08]	2008	MB	○	◑	○	○	●	○	◑	○	○
Chahadi, [Cha08]	2008	MB	○	○	○	○	◑	◑	◑	◑	◑
Müller, [Mül08]	2008	MB	○	●	○	○	◑	◑	◑	◑	◑
Tavassoli, [Tav08]	2008	IT	○	○	○	○	○	◑	◑	◑	◑
Bauer, [Ste08a]	2008	MB	○	○	◑	○	◑	○	○	◑	◑
VDI-RL 3694, [VDI08]	2008	MB	○	◑	◑	○	○	●	◑	◑	○
Volere, [Rob08]	2008	IT	◑	◑	◑	◑	◑	◑	●	◑	○

A.2 Eigenschaften von Anforderungen

In der Literatur (z.B. [Fra76] [IEE98a] [IEE98b] [Sch02] [Rup07]) finden sich eine Reihe von Eigenschaften, die das freigegebene Anforderungskollektiv aufweisen soll:

- konsistent
  Das Anforderungskollektiv soll eindeutig sein. Dazu sollen die Anforderungen voneinander unabhängig und insbesondere frei von Widersprüchen sein. Änderungen einer Anforderung sollen sich nicht auf andere Anforderungen auswirken können.

- minimal
  Das Anforderungskollektiv soll einen der Entwicklungsaufgabe angemessenen—möglichst minimalen—Umfang haben.

- modifizierbar
  Das Anforderungskollektiv soll über den gesamten Produktentwicklungsprozess verändert und erweitert werden können, so dass zu verschiedenen Zeitpunkten mit unterschiedlichen Versionen gearbeitet werden kann. Dazu muss der Satz aufgabenabhängig sortierbar sein und alle Teammitgliedern müssen gemeinsam darauf zugreifen können. Die Änderungen müssen verfolgbar sein.

- redundanzfrei
  Jede Anforderung soll nur einmal existieren und verschiedene Anforderungen sollen sich nicht überlappen.

- repräsentierbar
  Aus dem Anforderungskollektiv können Repräsentationen in spezifischen Sichten und Umfang erzeugt werden.

- strukturiert
  Das Anforderungskollektiv soll bezüglich den festgelegten Randbedingungen, Anwendungsbereich, Kontext und Vorgehen geeignet strukturiert sein. Insbesondere sollen die Anforderungen auf gleicher granularer Ebene vorliegen und Beziehungen zwischen den Anforderungen definiert sein.

- vollständig
  Das Anforderungskollektiv soll alle für die Produktentwicklung notwendigen Anforderungen enthalten.

Innerhalb des Anforderungskollektivs soll jede einzelne Anforderung diese Eigenschaften aufweisen (z.B. [Bid68] [Kic95] [IEE98b] [Sch02] [Rup07] [Tav08]):

- eindeutig
  Die Anforderung soll nur einen Gegenstand beschreiben, d.h. nicht verschiedene Elementaranforderungen zusammenfassen.

- gültig und aktuell
  Die Anforderung muss dem aktuellen Stand der Produktentwicklung entsprechen. Veraltete oder unkorrekte Anforderungen müssen als ungültig markiert werden.

- klar und unmissverständlich
  Die Anforderung soll so eindeutig formuliert sein, dass nur eine Interpretation möglich ist. Insbesondere sollen auch Entwickler aus unterschiedlichen Fachbereichen das gleiche unter einer Anforderung verstehen.

- klassifiziert
  Die Anforderung soll für eine bessere Übersichtlichkeit bestimmten—von der Entwicklungsaufgabe abhängigen—Klassen zugeordnet werden.

- konsistent
  Die Anforderung soll in sich konsistent sein.

- korrekt
  Die Anforderung muss richtig wiedergegeben sein, z.B. der angegebene Wertebereich muss stimmen.

- lösungsunabhängig
  Die Anforderung soll keine Lösungsrichtung vorgeben, sondern möglichst lösungsneutral formuliert sein.

- notwendig
  Die Anforderung muss für den Produktentwicklungsprozess notwendig und hilfreich sein. Unnötige Anforderungen vergrößern nur die Datenbasis ohne ein mehr an Informationen zu bieten. Durch die verschlechterte Übersicht wird auch die Lösungssuche eher verschlechtert denn verbessert.

- priorisiert
  Die Anforderung soll nach ihrer Priorität als Fest-, Mindest- oder Wunschforderung markiert sein.

- realisierbar
  Die Anforderung soll im Produkt und unter den übrigen Randbedingungen (z.B. Zeit-, Kostenrahmen) umsetzbar sein.

- verfolgbar
  Änderungen der Anforderung während der Produktentwicklung sollen verfolgbar und dadurch nachvollziehbar sein.

- verifizierbar
  Die Erfüllung bzw. der Erfüllungsgrad der Anforderung soll überprüf- und belegbar sein.

- vollständig
  Die Anforderung soll vollständig hinsichtlich der festgelegten Eigenschaften sein (z.B. Beschreibung, Werte, Autor, Quelle).

Anhang B

Anhang

B.1 Modellierung

Modelle der Produktentwicklung sollen die in Tabelle B.1 zusammengestellten allgemeinen Ziele erfüllen. Die Merkmale, durch die Modelle charakterisiert werden können (vgl. Tabelle B.2), haben einen Einfluss auf die Erfüllung der Ziele (vgl. Tabelle B.3). Je nachdem mit welchen Werten die Merkmale belegt werden, werden die Ziele unterschiedlich gut erfüllt.

Tabelle B.1: *Allgemeine Ziele, die von Modellen erfüllt werden sollen.*

	Analyse
Z.1	Die Modelldaten sollen in geeigneter Weise (z.B. durch geeignete Werkzeuge) aufbereitet, bereitgestellt und analysiert werden (nach [Qui00, S. 26]).
	Arbeitsaufwand
Z.2	Der notwendige Arbeitsaufwand zum Erlernen, zur Erstellung und zur Nutzung des Modells muss vertretbar und durch den erzielbaren Nutzen gerechtfertigt sein, z.B. intuitive grafische Modellierung (nach [Qui00, S. 27], [Gau08a, S. 65]).
	Dokumentation
Z.3	Das Modell soll die Modelldaten und getroffene Entscheidungen für eine spätere Weiterverwendung dokumentieren (nach [Gah04, S. 161]).
	Durchgängigkeit
Z.4	Das Modell soll mit fortlaufender Entwicklung und steigendem Informationssniveau kontinuierlich konkretisiert werden und durchgängig nutzbar sein (nach [Qui00, S. 27] [Gah04, S. 161] [Gau08a, S. 65]).
	Erweiterbarkeit
Z.5	Das Modell soll auf neue Randbedingungen (z.B. neue notwendige Partialmodelle) erweitert oder angepasst werden können (nach [Avg07, S. 40ff] [Gau08a, S. 65]).
	Flexibilität
Z.6	Das Modell soll flexibel auf neue Gegebenheiten des zu modellierenden Gegenstandes und des Prozesses eingehen können (nach [Gah04, S. 161] [Avg07, S. 40ff]).
	Ganzheitlich
Z.7	Das Modell soll den Modellierungsgegenstand zumindest bis auf Prinzipebene ganzheitlich betrachten (nach [Gau08a, S. 65]).
	Gleichberechtigung
Z.8	Die an der Entwicklung beteiligten Fachbereiche sollen gleichberechtigt in de Modellierung eingehen (nach [Gau08a, S. 65]).

Tabelle B.1: ***(Fortsetzung)*** *Allgemeine Ziele, die von Modellen erfüllt werden sollen.*

Z.9	**Konsistenz** Alle im Modell verwendeten Daten sollen frei von Widersprüchen und Redundanzen sein und das Modell ist logisch und semantisch richtig aufgebaut (nach [Qui00, S. 26] [Avg07, S. 40ff]).
Z.10	**Kooperation** Das Modell soll in der Lage sein mit weiteren Modellen (z.B. zur Entscheidungsunterstützung) zusammenzuwirken, indem z.B. Daten untereinander austauschbar sind (nach [Gah04, S. 161]).
Z.11	**Konzipierung** Das Modell soll in der Lage sein die Konzepterstellung zu unterstützen (nach [Gau08a, S. 65]).
Z.12	**Produktlebenslauf** Das Modell soll eine integrierte Betrachtung aller Produktlebensphasen ermöglichen (nach [Gah04, S. 161]).
Z.13	**Transparenz** Das Modell die Transparenz erhöhen, z.B. indem Zusammenhänge sichtbar werden und gleichzeitig soll die Komplexität beherrschbar bleiben ([Ort98] [Gah04, S. 161] [Avg07] [Gau08a, S. 65]).
Z.14	**Unsicherheiten** Die Größe von Unsicherheiten z.B. auf Grund geringer Informationen zu Entwicklungsbeginn sollen darstellbar sein (nach [Gah04, S. 161]).
Z.15	**Verfügbarkeit** Das Modell soll zu jedem Zeitpunkt in seiner aktuellen Version verfügbar sein.
Z.16	**Vollständigkeit** Das Modell soll im angelegten Betrachtungsrahmen vollständig in Bezug auf Ursache und Wirkung sein (nach [Gah04, S. 161]).

Tabelle B.2: *Allgemeine Merkmale, durch die Modelle charakterisiert werden können.*

M.1	**Abdeckung** Bereich des Modellierungsgegenstandes (Eigenschaften, Funktionalitäten), der durch das Modell abgedeckt wird (nach [Avg07, S. 40ff]).
M.2	**Abstraktionsgrad** Abstraktionsniveau, auf dem das Modell beschrieben wird.
M.3	**Aktualität** Das Modell gibt den aktuellen Stand des Modellierungsgegenstandes wider und kann bei Bedarf aktualisiert werden (nach [Avg07, S. 40ff]).
M.4	**Austauschbarkeit** Modellelemente lassen sich durch andere austauschen ohne das dadurch Einschränkungen in der Funktionalität oder den Eigenschaften auftreten (nach [Avg07, S. 40ff]).
M.5	**Autonomie** Das Modell überprüft sich selbst (nach [Avg07, S. 40ff]).
M.6	**Blickwinkel** Die Sichtweise aus der das Modell erstellt wurde und damit das Festlegen von zu vernachlässigen Elementen (nach [Avg07, S. 40ff]).

Tabelle B.2: ***(Fortsetzung)*** *Allgemeine Merkmale, durch die Modelle charakterisiert werden können.*

	Denkmuster
M.7	Das Modell ist auf Basis eines speziellen Denkmusters einer speziellen Fachdisziplin erstellt worden, z.B. Objektorientierung (nach [Avg07, S. 40ff]).
	Dimension
M.8	Anzahl der (z.B. physikalischen) Dimensionen, die das Modell abbildet (nach [Avg07, S. 40ff]).
	Funktionalität
M.9	Anzahl der unterschiedlichen Funktionalitäten, die im Modell implementiert sind [Avg07, S. 40ff].
	Genauigkeit
M.10	Das Modell gibt den Modellierungsgegenstand ohne große Abweichungen zur Realität wieder und liefert auch bei mehrfacher Anwendung die immer gleichen Ergebnisse (nach [Avg07, S. 40ff]).
	Größe
M.11	Größe der benötigten Informationsspeicher, z.B. Anzahl MB an Datenspeicher bei Softwaremodellen (nach [Avg07, S. 40ff]).
	Modularität
M.12	Modularität der Daten in Bezug auf, Individuelle Zusammenstellung, Auswertung und Analyse der Daten unter Nutzung der Erfahrung (nach [Qui00, S. 27]).
	Plattform
M.13	Zur Modellierung notwendiges Material, z.B. Papier, Computerhard- und -software (nach [Avg07, S. 40ff]).
	Portabilität
M.14	Das Modell kann mit geringem Aufwand bzw. Änderungen auf einer anderen Plattform verwendet werden (nach [Avg07, S. 40ff]).
	Repräsentation
M.15	Die Art der Repräsentation des Modellierungsgegenstandes im Modell, z.B. textuell, grafisch (nach [Ull92, S. 28]).
	Robustheit
M.16	Das Modell ist in der Lage trotz (unvorhergesehener) negativer Einwirkungen die vorgegebenen Eigenschaften und Funktionalitäten einzuhalten (nach [Avg07, S. 40ff]).
	Skalierbarkeit
M.17	Das Modell oder Teile des Modells können in mehreren Instanzen parallel eingesetzt werden (nach [Avg07, S. 40ff]).
	Unabhängigkeit
M.18	Das Modell hat wenige und nur schwache Beziehungen und Abhängigkeiten zu den Elementen seiner Umgebung (nach [Avg07, S. 40ff]).
	Universalität
M.19	Das Modell kann für eine Anzahl unterschiedlicher Zwecke eingesetzt werden (nach [Avg07, S. 40ff]).
	Wiederverwendbar
M.20	Das Modell oder Modellelemente können in anderen Zusammenhängen wiederverwendet werden (nach [Avg07, S. 40ff]).
	Zeitabhängigkeit
M.21	Das Modell lässt sich abhängig von der Phase im Produktentwicklungsprozess in Bezug auf Repräsentation und Verhalten verändern (nach [Avg07, S. 40ff]).
	Zusammensetzung
M.22	Das Modell ist als ein Ganzes anzusehen oder ein aus mehreren Teilen zusammengesetztes Verbundmodell, dass in verschiedene kleinere Untermodelle zu teilen ist (nach [Avg07, S. 40ff]).

Tabelle B.3: *Gegenüberstellung von Modellierungszielen (Z vgl. Tabelle B.1) und Modellmerkmalen (M vgl. Tabelle B.2) als Beeinflussungsmatrix (○ keine Beeinflussung, ● das Modellierungsmerkmal beeinflusst die Zielerreichung).*

	Z.1	Z.2	Z.3	Z.4	Z.5	Z.6	Z.7	Z.8	Z.9	Z.10	Z.11	Z.12	Z.13	Z.14	Z.15	Z.16
M.1	○	●	○	●	○	●	●	●	○	○	○	●	○	●	○	●
M.2	●	●	●	○	●	●	●	●	●	●	●	●	●	●	○	●
M.3	○	○	●	○	○	○	○	○	●	●	○	●	○	●	●	●
M.4	○	●	○	○	●	●	○	○	●	●	○	○	○	○	○	○
M.5	○	●	○	○	○	○	○	○	●	○	○	○	○	○	○	○
M.6	○	○	○	○	○	●	●	●	○	●	●	●	●	○	○	○
M.7	○	○	○	○	○	●	●	●	○	○	●	○	○	○	○	○
M.8	○	●	○	○	○	○	○	○	○	○	●	○	○	○	○	●
M.9	●	●	○	○	○	○	○	○	○	○	●	○	○	○	○	○
M.10	○	●	○	○	○	○	○	○	○	●	○	○	○	●	○	○
M.11	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	●
M.12	○	○	○	○	●	●	○	○	○	●	○	●	○	○	○	○
M.13	○	●	○	○	●	○	○	○	○	●	○	○	○	○	●	○
M.14	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○
M.15	●	●	●	○	○	○	●	●	○	●	●	○	●	○	○	○
M.16	○	○	○	○	○	○	○	○	●	○	○	○	○	●	○	○
M.17	●	●	○	○	●	○	●	○	○	○	○	●	○	○	○	●
M.18	○	○	○	○	●	●	●	○	○	●	○	○	○	○	○	○
M.19	○	○	○	○	●	●	●	●	○	●	○	●	○	○	○	○
M.20	○	●	○	○	●	○	○	○	○	●	○	●	○	○	○	○
M.21	○	○	○	●	○	○	●	○	○	○	●	●	○	○	○	○
M.22	○	○	○	○	●	○	○	○	○	○	○	●	○	○	○	○

Tabelle B.4: *Analyse bekannter Modellierungssprachen [Gau08a, S. 65] (● voll erfüllt, ◐ teilweise erfüllt, ○ nicht erfüllt).*

Anforderungen / Ansätze	Ganzheitliche Abbildung der Prinziplösung	Intuitive grafische Modellierung	Unterstützung der Konzipierung	Gleichberechtigung der Domänen	Beherrschung der Komplexität	Durchgängigkeit	Erweiterbarkeit
Axiomatic Design	◐	●	◐	●	◐	●	●
Funktionsorientierte Spezifikation mechatronischer Systeme	◐	●	◐	●	◐	○	●
Funktionsorientiertes Entwerfen	◐	●	◐	○	◐	●	◐
Elementmodell—Wirkflächen- und Leitstützstrukturen	◐	●	◐	○	◐	○	◐
Bondgraphen	◐	●	◐	○	◐	○	◐
Blockdiagramme	◐	●	◐	○	◐	○	◐
VHDL-AMS	◐	●	◐	◐	◐	◐	◐
Modelica	◐	●	◐	●	◐	◐	◐
UML—Unified Modeling Language	◐	●	◐	○	◐	●	◐
Mechatronic UML	◐	●	◐	◐	◐	◐	◐
SysML—Systems Modeling Language	◐	●	◐	◐	●	●	◐
Verhaltensspezifikation nach Schön mit AssMePro	◐	●	◐	◐	◐	◐	◐
Schemebuilder Mechatronics	◐	●	◐	◐	◐	◐	◐
ModCoDe—Modeling System for Conceptual design	◐	●	◐	◐	●	●	◐
MeSoMod—Mechasoft Modeller	◐	●	◐	○	●	●	◐
20-sim	◐	●	◐	◐	◐	◐	◐
CARTRONIC	◐	●	◐	◐	◐	◐	◐

B.2 Mögliche gegenseitige Beeinflussung der Partialmodelle

Abb. 3.3 gibt an, welche der betrachteten Partialmodelle miteinander in Beziehung stehen. Im Folgenden werden diese Beeinflussungsmöglichkeiten genauer beschrieben. Die Beschreibung folgt der Darstellung in Abb. 3.3 von oben nach unten:

Stakeholdermodell

- Produktlebenslauf
  Stakeholder sind in verschiedenen Phasen des Produktlebenslaufs involviert. Beispielsweise sind Monteure am Aufbau einer Maschine beteiligt und Maschinenbediener bei ihrer Nutzung.

- Produktumgebung
  Im Stakeholdermodell werden die Personen festgehalten, die z.B. Auskunft über mögliche Auswirkungen von Nachbarsystemen auf das Produkt geben können, weil sie z.B. Bediener oder Entwickler der zu betrachtenden Nachbarsysteme sind.

- Systemidee
  Die Objekte der Systemidee entstammen einer Quelle, die häufig Teil des Stakeholdermodells ist. Beispielsweise ist das Objekt der Systemidee „*hohe Flexibilität*“ vom Betreiber motiviert. Weitere Zusammenhänge ergeben sich durch die Autoren bzw. Verantwortlichen der Objekte.

- Ziele
  Die Ziele entstammen einer Quelle, die häufig Teil des Stakeholdermodells ist. Beispielsweise ist das Ziel „*hohe Nutzlast erzielen*“ vom Betreiber motiviert. Weitere Zusammenhänge ergeben sich durch die Autoren bzw. Verantwortlichen der Objekte.

- Anforderungsmodell
  Die Anforderungen entstammen einer Quelle, die häufig Teil des Stakeholdermodells ist. Beispielsweise ist die Anforderung „*einfache Bedienung ermöglichen*“ aus einem Interview mit einem Maschinenbediener dokumentiert. Weitere Zusammenhänge ergeben sich durch die Autoren bzw. Verantwortlichen der Objekte.

- Kostenmodell
  Für bestimmte Kosten können verantwortliche Personen festgelegt werden, die im Stakeholdermodell dokumentiert werden. Außerdem können Stakeholder unter Angabe von Lohnkosten im Kostenmodell „verrechnet“ werden (z.B. Durchschnittssatz für Maschinenbediener 80 Euro/Stunde).

- Testmodell
  Die Mitglieder der Testteams werden im Stakeholdermodell dokumentiert.

Produktlebenslauf

- Produktumgebung
  Einige Objekte der Produktumgebung sind nur in bestimmten Produktlebenslaufphasen zu berücksichtigen. Beispielsweise sind Wartungsgeräte nur in Wartungsphasen zu berücksichtigen.

- Anforderungsmodell
  Die im Produktlebenslaufmodell abgelegten Anwendungsfälle tragen häufig zur Verfeinerung von Anforderungen bei.

- Funktionsmodell
  Die im Produktlebenslaufmodell abgelegten Anwendungsfälle können ggf. im Funktionsmodell konkretisiert werden. Beispielsweise kann der Anwendungfall „*Sitz auf Benutzergröße einstellen*“ entsprechend in eine Funktionsstruktur umgesetzt werden.

- Kostenmodell
  Wird das Kostenmodell z.B. als Prozesskostenrechnung umgesetzt, so können die relevanten Anwendungsfälle als Prozesse im Kostenmodell verwendet werden.

- Testmodell
  Die Anwendungsfälle des Produktlebenslaufmodells können genutzt werden, um Testfälle für die Testkriterien zu entwickeln, die sich aus den entsprechenden Anforderungen ergeben.

Produktumgebung

- Anforderungsmodell
  Aus den Objekten der Produktumgebung ergeben sich häufig Anforderungen. Beispielsweise müssen Schnittstellen zu den Nachbarsystemen eingehalten (z.B. Achshöhe der Ausgangswelle eines Getriebes) oder Störeffekte berücksichtigt werden (z.B. lokal hohe Temperatur durch einen Antrieb).

- Verhaltensmodell
  Die Produktumgebung beeinflusst das Verhalten durch externe Eingangsgrößen bzw. Ereignisse, die bestimmte Zustandsübergänge auslösen.

- Testmodell
  Das Produktumgebungsmodell bildet die Vorlage für die Modellierung der

Testumgebung. Beispielsweise kann eine Produktumgebung virtuell im Testmodell abgebildet und ggf. für unterschiedliche Varianten angepasst werden (z.B. Hardware-in-the-Loop).

Systemidee

- Ziele
  Ziele sind unter anderem Konkretisierungen der Objekte der Systemidee. Beispielsweise kann durch die Erfüllung des Ziels „*einfache Rekonfiguration des Robotersystems erzielen*“ ebenfalls die Systemidee „*breiten Anwendungsbereich mit einem Handhabungssystem abdecken*“ zumindest teilweise erfüllt werden.

- Anforderungsmodell
  Die Erfüllung einer Anforderung kann (über die Erfüllung von Zielen) zur Erfüllung von Objekten der Systemidee beitragen.

Ziele

- Anforderungsmodell
  Die Erfüllung einer Anforderung leistet häufig einen Beitrag zur Erfüllung eines Ziels. Beispielsweise trägt die Anforderung „*Einheitliche Benutzungsoberfläche der Bedienelemente bei allen Varianten*“ einen Teil zur Erfüllung des Ziels „*kurze notwendige Einarbeitungszeit erzielen*“ bei.

- Kostenmodell
  Im Zielmodell werden Zielkosten definiert, die z.B. mit den Kostenschätzungen des Kostenmodells verglichen werden können (vgl. Target Costing (TC)).

Anforderungsmodell

- Funktionsmodell
  Viele der relevanten Anforderungen der frühen Phasen sind funktionale Anforderungen. Diese bilden die Grundlage für die Erstellung einer Funktionsstruktur.

- Strukturmodell
  Bei der Entwicklung von Systemen können bestimmte Anforderungen häufig einzelnen Systemkomponenten zugeordnet werden (z.B. notwendiger Gelenkfreiheitsgrad dem entsprechenden Gelenk). Die Komponenten werden im Strukturmodell abgebildet.

- Verhaltensmodell
  Anforderungen, die sich auf das Systemverhalten beziehen (z.B. geforderte maximale Reaktionszeit auf ein Ereignis), werden mit den entsprechenden Objekten des Verhaltensmodells (z.B. Aktivitäten) verknüpft.

- Kostenmodell
  Anforderungen können Konkretisierungen der Kostenziele enthalten und werden mit den entsprechenden Objekten des Kostenmodells verbunden.

- Testmodell
  Anforderungen müssen überprüfbar sein, daher bilden sie die Grundlage für die Definition von Testkriterien. Anforderungen werden dazu umformuliert und ggf. die genauen Abnahmekriterien und Prüfbedingungen festgelegt.

Funktionsmodell

- Strukturmodell
  Funktionen können im Strukturmodell bestimmten „Funktionsträgern“ zugeordnet werden. Beispielsweise erfüllt die kinematische Struktur eines Roboters die Funktion „*Bewegung wandeln*“.

- Verhaltensmodell
  Das Verhaltensmodell ist häufig eine Erweiterung des Funktionsmodells. Beispielsweise werden durch ein Ereignis dynamisch verschiedene Aktivitäten ausgeführt, die jeweils durch Funktionen erfüllt werden.

- Testmodell
  Das Funktionsmodell kann häufig bereits für erste Funktionstest verwendet werden, z.B. indem bei festgelegten Parametern die Ausgangsgrößen zu bestimmten Eingangsgrößen errechnet und mit den Sollwerten verglichen werden (vgl. SFS).

Strukturmodell

- Verhaltensmodell
  Bestimmte Zustände bzw. Aktivitäten des Verhaltensmodells lassen sich auf Objekte des Strukturmodells abbilden. Beispielsweise werden Aktivitäten zur Darstellung einer Benutzungsoberfläche auf der Komponente „Grafikkarte“ ausgeführt.

- Testmodell
  Das Strukturmodell kann bereits auf abstrakter Ebene genutzt werden, um wesentliche Testkriterien zu überprüfen bzw. wenigstens Varianten auszuschließen. Beispielsweise kann durch die im Strukturmodell angegebene Anordnung der Elemente bereits ein Bauraumbedarf abgeschätzt werden, der mit Sollwerten verglichen werden kann.

Verhaltensmodell

- Testmodell
  Auf Basis des Verhaltensmodells können bereits teilweise Testfälle überprüft werden. Damit kann sichergestellt werden, dass zumindest das Modell bereits das gewünschte Verhalten für bestimmte Ereignisse zeigt. Für die Software- und Elektronikentwicklung existieren hier eine große Anzahl von Verfahren zur automatischen Testfallgenerierung.

Kostenmodell

- Testmodell
  Die Einhaltung von Kostenzielen muss im Projektverlauf kontinuierlich überprüft werden. Dazu werden Objekte des Kostenmodells mit dem Testmodell verknüpft.

Anhang C

Anhang

C.1 Anforderungsmanagement in der Vorlesung RAO

Für die Auswertung der Untersuchungen im Rahmen der Vorlesung „Rechnerunterstütztes Auslegen und Optimieren“ (RAO) wurde ein Fragebogen entwickelt. Dieser wurde von allen Teilnehmern am Ende der Vorlesung anonym ausgefüllt und abgegeben. Anschließend wurden die Angaben ausgewertet. Der Fragebogen ist in Abb. C.1 abgebildet.

Technische Universität Braunschweig
Institut für Konstruktionstechnik
Prof. Dr.-Ing. Hans-Joachim Franke

5.12.2008
Ste

Fragebogen – RAO im WiSe 08/09

Bitte füllen Sie den Fragebogen entsprechend ihren subjektiven Erfahrungen aus. Markieren Sie Zutreffendes bitte so ☒ und verwenden Sie Kugelschreiber oder Filzstift. Haben Sie darüber hinaus Kommentare, Anmerkungen und Anregungen, so verwenden Sie bitte die Rückseite.

Zur Person

In welchem Fachsemester studieren Sie? ☐ 5. ☐ 7. ☐ 9. ☐ 11. ☐ 13. und mehr

In welchem Studiengang studieren Sie? ☐ Maschinenbau ☐ Wi.-Ing Maschinenbau ☐ _____

Haben Sie an der Zusatzqualifikation „Anforderungsmanagement“ teilgenommen? ☐ ja ☐ nein

Produktentwicklung

Für wie wichtig halten Sie die Klärung von Anforderungen?	wichtig ☐ ☐ ☐ ☐ ☐	unwichtig	☐ k.A.
Waren Ihnen die Entwicklungsziele während der PE[1] klar?	immer ☐ ☐ ☐ ☐ ☐	selten	☐ k.A.
Hat die Aufgabenklärung bei der PE geholfen?	ja ☐ ☐ ☐ ☐ ☐	nein	☐ k.A.
Haben Sie das Anforderungsmodell während der PE laufend ergänzt?	immer ☐ ☐ ☐ ☐ ☐	nie	☐ k.A.
Wieviel Zeit haben Sie ungefähr in die Aufgabenklärung investiert?	_____	Stunden	☐ k.A.
Wieviel Zeit haben Sie ungefähr in die Auslegung[2] investiert?	_____	Stunden	☐ k.A.
Wieviel Zeit haben Sie ungefähr in die Gestaltung[3] investiert?	_____	Stunden	☐ k.A.
Wieviele Iterationen mussten Sie ungefähr durchführen?	_____	Iterationen	☐ k.A.

Anforderungsmanagement mit SysML für Teilnehmer der Zusatzqualifikation

Wie Beurteilen Sie die Erlernbarkeit der Anforderungsmodellierung?	einfach ☐ ☐ ☐ ☐ ☐	schwierig	☐ k.A.
Konnten Sie sich schnell in die Anforderungsmodellierung einarbeiten?	schnell ☐ ☐ ☐ ☐ ☐	langsam	☐ k.A.
Wie erschien Ihnen die Bedienbarkeit von ARTiSAN Studio v7?	intuitiv ☐ ☐ ☐ ☐ ☐	kompliziert	☐ k.A.
Half Ihnen die vorgestellte Methodik bei der Modellerstellung?	ja ☐ ☐ ☐ ☐ ☐	nein	☐ k.A.
Half Ihnen die Modellierung wichtige Zusammenhänge zu erkennen?	ja ☐ ☐ ☐ ☐ ☐	nein	☐ k.A.
Birgt ein Anforderungsmodell wichtige Zusatzinformationen gegenüber einer Anforderungsliste?	ja ☐ ☐ ☐ ☐ ☐	nein	☐ k.A.

Die Anforderungsmodellierung hilft Ihrer Meinung nach bei (Mehrfachnennungen erlaubt):

☐ Neuentwicklungen ☐ Anpassentwicklungen ☐ Variantenentwicklungen ☐ _____

[1] Produktentwicklung
[2] Aufgabenteile mit Excel, Maple, eAssistant-Berechnungen
[3] Gestaltung in CAD, Erstellung der Fertigungsunterlagen

Abbildung C.1: *Fragebogen zur Auswertung des Anforderungsmanagements im Rahmen der Vorlesung Rechnerunterstütztes Auslegen und Optimieren im Wintersemester 2008/2009*

Berichte aus dem Institut für Konstruktionstechnik

Nr. 37 **Barrenscheen, Jörg:** Die systematische Ausnutzung von Symmetrieeigenschaften beim Konstruieren. Dissertation TU Braunschweig 1990

Nr. 38 **Drebing, Uwe:** Zur Metrik der Merkmalsbeschreibung für Produktdarstellende Modelle beim Konstruieren. Dissertation TU Braunschweig 1991

Nr. 39 **Weigel, Klaus D.:** Entwicklung einer modularen Systemarchitektur für die rechnerintegrierte Produktgestaltung. Dissertation TU Braunschweig 1991

Nr. 40 **Steffens, Ralf:** Die Profilsteigungsfunktion, ein neuer Weg zur analytischen Bestimmung und Optimierung allgemeiner Profilflankenpaarungen. Dissertation TU Braunschweig 1993

Nr. 41 Verschiedene Autoren: Veröffentlichungen des Instituts für Konstruktionslehre, Maschinen- und Feinwerkelemente 1989 bis 1993, 1993

Nr. 42 **Böwer, Gunnar:** Untersuchung der konzeptionellen Erweiterungsmöglichkeiten von CAD-Systemen am Beispiel der rechnerunterstützten Bemaßungsanalyse und Toleranzberechnung. Dissertation TU Braunschweig 1993

Nr. 43 **Speckhahn, Hermann:** Systeme zur flexibel konfigurierbaren Informationsbereitstellung für die Konstruktion. Dissertation TU Braunschweig 1995

Nr. 44 **Kickermann, Heiner:** Rechnerunterstützte Verarbeitung von Anforderungen im methodischen Konstruktionsprozeß. Dissertation TU Braunschweig 1995

Nr. 45 **Bielfeldt, Uwe:** Ein Beitrag zur konstruktionsmethodischen Entwicklung von Pumpen für die Biotechnologie. Dissertation TU Braunschweig 1996

Nr. 46 **Hacker, Günther:** Untersuchungen zur methodischen Gestaltung von Maschinengehäusen. Dissertation TU Braunschweig 1996, Göttingen: Cuvillier Verlag 1996. ISBN 3-89588-609-2

Nr. 47 **Fischer, Rolf:** Product Design based on HyperTrees. Dissertation TU Braunschweig 1996

Nr. 48 **Renken, Martin:** Nutzung recyclingorientierter Bewertungskriterien während des Konstruierens. Dissertation TU Braunschweig 1996, Göttingen: Cuvillier Verlag 1996. ISBN 3-89588-555-X

Nr. 49 **Schulz, Achim:** Systeme zur Rechnerunterstützung des funktionsorientierten Grobentwurfs. Dissertation TU Braunschweig 1996

Nr. 50 **Tsai, Shyi-Jeng:** Vereinheitlichtes System evolventischer Zahnräder. - Auslegung von Zylindrischen, Konischen, Kronen- und Torusrädern. Dissertation TU Braunschweig 1997

Nr. 51 **Peters, Michael:** Kommunikationssystem rechnerunterstützter Konstruktionswerkzeuge. Dissertation TU Braunschweig 1998

Nr. 52 **Jeschke, Andrea:** Beitrag zur wirtschaftlichen Bewertung von Standardisierungsmaßnahmen in der Einzel- und Kleinserienfertigung durch die Konstruktion. Dissertation TU Braunschweig 1997

Nr. 53 **Lachmayer, Roland:** Temperatur- und wärmeflußgerechte Gestaltung axialer Gleitringdichtungen. Dissertation TU Braunschweig 1997, Aachen: Shaker Verlag 1998. ISBN 3-8265-3513-8

Nr. 54 **Döllner, Gernot:** Konzipierung und Anwendung von Maßnahmen zur Verkürzung der Produktentwicklungszeit am Beispiel der Aggregateentwicklung. Dissertation TU Braunschweig 1997

Nr. 55 **Kaletka, Ingo:** Zielgerichtetes Entwickeln im Methodischen Konstruktionsprozeß durch Verwendung ganzheitlicher Modelle. Dissertation TU Braunschweig 1999, Göttingen: Cuvillier Verlag 1999. ISBN 3-89712-386-X

Nr. 56 **Franke, H.-J., Krusche, T., Mette, M. (Hrsg.):** Konstruktionsmethodik - Quo vadis? Symposium anlässlich des 80. Geburtstags von Prof. Dr.-Ing. Karlheinz Roth, Aachen: Shaker Verlag 1999. ISBN 3-8265-4916-3

Nr. 57 **Lippardt, Sven:** Gezielte Förderung der Kreativität durch bildliche Produktmodelle. Dissertation TU Braunschweig 2000, Fortschritt-Berichte VDI Reihe 1 Nr. 325, Düsseldorf: VDI-Verlag, 2000. ISBN 3-18-332501-2

Nr. 58 **Fritsch, Joachim:** Erhöhung der Betriebssicherheit vollkeramischer Wälzlager unter vollständiger Umströmung mit niedrigviskosem Medium. Dissertation TU Braunschweig 2000. Aachen: Shaker Verlag 2000. ISBN 3-8265-7456-7

Nr. 59 **Kramer, Manfred:** Systematische Werkstoffauswahl im Konstruktionsprozeß am Beispiel von Kunststoffbauteilen im Automobilbau. Dissertation TU Braunschweig 2000

Nr. 60 **Krusche, Thomas:** Strukturierung von Anforderungen für einen effizienten und effektiven Konstruktionsprozeß. Dissertation TU Braunschweig 2000. Aachen: Verlag Mainz 2000. ISBN 3-89653-796-2

Nr. 61 **Lux, Stefan:** Entwicklung rechnerunterstützter Angebotssysteme mit generischen Methoden. Dissertation TU Braunschweig 2001. Aachen: Verlag Mainz 2001. ISBN 3-89653-856-X

Nr. 62 **Hagedorn, Uwe:** Reibungsverluste von vollständig mit niedrigviskosem Medium umströmten Wälzlagern. Dissertation TU Braunschweig 2001. Aachen: Shaker Verlag 2001. ISBN 3-8265-9419-3

Nr. 63 **Brey, Marco:** Konfiguration und Gestaltung mit Constraintsystemen. Dissertation TU Braunschweig 2003. Göttingen: Cuvillier Verlag 2003. ISBN 3-89873-643-1

Nr. 64 **Köberlein, Steffen:** Systematische wettbewerbsorientierte Produktentwicklung. Dissertation TU Braunschweig 2003. Göttingen: Cuvillier Verlag 2003. ISBN 3-89873-922-8

Nr. 65 **Firchau, Norman L.:** Variantenoptimierende Produktgestaltung. Dissertation TU Braunschweig 2003. Göttingen: Cuvillier Verlag 2003. ISBN 3-89873-934-1

Nr. 66 **Pini, P., Germer, C.** (Hrsg.): Konstruktionsmethodik in der Praxis - Einsatzmöglichkeiten und Grenzen. Kolloquium anlässlich des 60. Geburtstags von Prof. Dr.-Ing. H.-J. Franke, Braunschweig 2004

Nr. 67 **Otremba, Robert:** Systematische Entwicklung von Gelenken für Parallelroboter. Dissertation TU Braunschweig 2005. Berlin: Logos Verlag 2005. ISBN 3-8325-0811-2

Nr. 68 **Germer, Christoph:** Interdisziplinäres Toleranzmanagement. Dissertation TU Braunschweig 2005. Berlin: Logos Verlag 2005. ISBN 3-8325-0954-2

Nr. 69 **Kroker, Jens:** Schnittstellensystematik für modulare Fahrzeugkarosserien. Dissertation TU Braunschweig 2005. Berlin: Logos Verlag 2005. ISBN 3-8325-1096-6

Nr. 70 **Wandelt, Dennis:** Modellierung von mehrstufigen Fertigungsprozessen zur mehrdimensionalen Toleranzanalyse und -synthese. Dissertation TU Braunschweig 2007. Berlin: Logos Verlag 2007. ISBN 978-3-8325-1638-3

Nr. 71 **Deimel, Markus:** Ähnlichkeitskennzahlen zur systematischen Synthese, Beurteilung und Optimierung von Konstruktionslösungen. Dissertation TU Braunschweig 2007. Düsseldorf: VDI Verlag GmbH 2007. ISBN 978-3-18-339801-0

Nr. 72 **Koschorrek, Ralph:** Systematisches Konzipieren mittels Ähnlichkeitsmethoden am Beispiel von PKW-Karosserien. Dissertation TU Braunschweig 2007. Berlin: Logos Verlag 2007. ISBN 978-3-8325-1784-7

Nr. 73 **Löffler, Stefan:** Anwenden bionischer Konstruktionsprinzipe in der Produktentwicklung. Dissertation TU Braunschweig 2009. Berlin: Logos Verlag 2009. ISBN 978-3-8325-2154-7

Nr. 74 **Haupt, U., Sánchez Ruelas, J.G. (Hrsg.):** Konstruktionsmethodik für Fahrzeugkonzepte. Kolloquium anlässlich des Ausscheidens von Prof. Dr.-Ing. H.-J. Franke aus dem aktiven Dienst und Übergabe des Instituts an Prof. Dr.-Ing. T. Vietor, Braunschweig: ITS Niedersachsen 2010. ISBN 978-3-937655-22-2

Die Berichte sind (wenn kein Verlag angegeben ist) zu beziehen vom:

Institut für Konstruktionstechnik
Technische Universität Braunschweig
Langer Kamp 8
38106 Braunschweig

Technische Universität Braunschweig
Institut für Konstruktionstechnik
Prof. Dr.-Ing. Thomas Vietor
Langer Kamp 8
38106 Braunschweig
Telefon +49 531 391-3343
Telefax +49 531 391-4572
ikt@tu-braunschweig.de
www.ikt.tu-braunschweig.de